QUAECUMQUE VERA

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: RANDY GOEBEL

TITLE OF THESIS: ORGANIZING FACTUAL KNOWLEDGE IN A SEMANTIC NETWORK

DEGREE FOR WHICH THESIS WAS PRESENTED: MASTER OF SCIENCE

YEAR THIS DEGREE WAS GRANTED: 1977

THE UNIVERSITY OF ALBERTA

ORGANIZING FACTUAL KNOWLEDGE IN A SEMANTIC NETWORK

by

Randy Goebel

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON, ALBERTA

FALL, 1977

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled "Organizing Factual Knowledge in a Semantic Network", submitted by Randy Goebel in partial fulfilment of the requirements for the degree of Master of Science.

To Me

ABSTRACT

Semantic networks have provided a powerful formalism for representing declarative machine knowledge, but they have lacked the ability to provide an organizational structure which can facilitate the efficient access of concept-based facts relevant to an arbitrary query.

As a preliminary step to developing an organization for semantic network propositions, a probability distribution of truth values is proposed as a tool for representing the credibility of proposition, and as a prerequisite for defining the concept of a fuzzy topic predicate. Fuzzy topic predicates form the basis for the development of a topic hierarchy organization which can be super-imposed on a semantic network data base in order to classify topically-related propositions into fuzzy categories.

The Symbol-Mapping problem is described as the problem of providing fast access to propositions relevant to a query, and the advantages of a topic hierarchy solution over existing solutions (e.g. an IS-A hierarchy) are discussed.

ACKNOWLEDGEMENTS

I will always owe an intellectual debt to Len Schubert. His patience, understanding, and guidance led me to the frontiers of my own understanding.

I am indebted to Kenj McDonell for his moral and technical support without which I could have not succeeded in producing this thesis.

Many thanks are due Steve Sutphen, John Demco, and Alan Covington for their patience in developing and maintaining much of the system software for which I found the need. Steve deserves special thanks for his super-human efforts in maintaining the hardware. These guys are truly Computing Science Wizards.

I would like to thank my examiners, Jeff Sampson, Kelly Wilson, and John Hogan for their insightful comments, questions, and suggestions about both the content and the style of the final draft.

This research was supported by the National Research Council of Canada under Operating Grant A8818.

TABLE OF CONTENTS

Chapter Page

TABLE OF CONTENTS (continued)

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

> My own belief is that many philosophical difficulties and controversies arise from an insufficient realization of the difference between different kinds of knowledge, and of the vagueness and uncertainty that characterizes most of what we believe ourselves to know (Russell, 1948, page 421).

1.1 Prologue

If a machine is ever to behave in an intelligent manner it must know something about the world in which it lives. The question 'how can a machine know anything about its environment?' has been the subject of much concern in the field of Artificial Intelligence. Most approaches to machine theories of knowledge have been consisted of a representational schema and a utilizational schema, which together provide an artificial system with the ability to represent and use knowledge to suit its purpose. In most knowledge theories this distinction has remained transparent, although basic controversies and biases (e.g. McCarthy & Hayes 1969, Winograd 1975) have arisen from the discussion of the differences between representing and

using knowledge.

The development of actual knowledge models (e.g. Schank et al. 1975, Bobrow et al. 1977) also exhibits this duality, with representational and utilizational components being more or less visible at the completion of each of their developmental cycles. Only a relatively cursory analysis of what transpires between these development cycles is necessary to discover the interdependence of these two major components. The nature of this interface is often regarded as being either representational or utilizational, depending on the interpreter's bias. Recent trends in knowledge research (e.g. scripts, frames, schemata, scenarios) indicate that this interface is perhaps another fairly distinct component, an organizational schema. This thesis is organized into representational, organizational, and utilizational components in the hope that such a three-part theme will provide a stimulating framework for discussion of a simple model of machine knowledge organization.

1.2 Focus

Semantic networks have provided an invaluable tool for investigating the structure and substance of machine knowledge. Their popular acceptance has been swift (less than a decade), with an impressive array of variants being produced (e.g. Quillian 1968, Schank 1972, Simmons 1973, Rumelhart et al. 1975, Schubert 1975). The early role of semantic networks (e.g. Quillian 1969) was generally one of

providing domain specific 'semantic memory' for systems whose exhibited behaviour was of most concern. For the most part, the pragmatic component of semantic network usage was buried in the programming of such systems, because of the major emphasis placed on developing the exhibited behaviour of such systems. These efforts to develop the visibly obvious intelligent behaviour of programmed systems has lead to the belief that knowledge and its use plays a vital role in intelligent systems. The recent development of organizational theories support this hypothesis, and the subsequent need for representing and using larger and more complex domains of real world knowledge has put explicit focus on the pragmatic component of organizing and using semantic network knowledge models.

Semantic networks have provided a powerful formalism for representing declarative machine knowledge, but they have lacked the ability to provide an organizational structure which can facilitate the efficient access of concept-based facts relevant to an arbitrary query. As a preliminary step to developing an organization for semantic network propositions, a probability distribution of truth values is proposed as a tool for representing the credibility of propositions, and as a prerequisite for defining the concept of a fuzzy topic predicate. Fuzzy topic predicates form the basis for the development of a topic hierarchy organization which can be super-imposed on a semantic network data base in order to classify topically-

related propositions into fuzzy categories. The Symbol-Mapping problem is described as the problem of providing fast access to propositions relevant to a query, and the advantages of a topic hierarchy solution over existing solutions (e.g. an IS-A hierarchy) are discussed.

1.3 Background Influences

The single most direct influence on this research has been semantic networks themselves, or rather their often discussed[1] tendency to suggest methods and structures which can be used to augment their own expressive power and ease their computational use. In this case, this influence is largely attributable to the succinct notation of Schubert (1975), subsequent uses of the notation in Cercone & Schubert (1975) and Cercone (1975), as well as the work of Schank (1972), Rumelhart et al. (1975), Quillian (1968,1969), Winston (1970), and the philosophy of Quine (1958, 1960, 1969, 1971).

Schank & Rieger (1974), LeFaivre (1974, 1977) and Zadeh (1965, 1974) have provided a background for the development of a representation for conceptual vagueness and propositional credibility. The development of scripts (Abelson 1973, Schank 1975, Abelson 1975, Abelson & Schank 1975), frames (Minsky 1974, Kuipers 1975), schemata (Bobrow & Norman 1975), demons (Charniak 1972, 1973), and scenarios

---

[1] for instance, see Quillian (1968), Schubert (1975).

(Mylopolous et al. 1977) has provided the impetus for investigating knowledge organizations.

The actual development of a semantic network organization has been spurred by the presentation of the Symbol-Mapping problem by Fahlman (1975), and by subsequent discussions of it by McDermott (1975) and Moore (1975). The work of Bobrow & Winograd (1976), Bobrow et al. (1977) and Charniak (1972,1976) has provided valuable insights into the problems of representing, organizing, and using machine knowledge.

1.4 Preview

Chapters 2, 3, and 4 constitute the main body of this thesis, and their organization reflects the three part theme presented in the Prologue. Chapter 2 deals with representing knowledge, Chapter 3 with organizing the knowledge being represented, and Chapter 4 discusses the application and use of proposed representational and organizational concepts in a propositional data base model.

Chapter 2 describes the use of semantic networks for representing knowledge, and presents the notation of Schubert (1975) as a suitable candidate. A short description of his original notation is followed by a brief presentation of previous ideas about representing vagueness and uncertainty within knowledge systems, then a new idea for representing both the vagueness of concepts and the

credibility of propositions is proposed as an extension to the existing notation.

Chapter 3 describes the Symbol-Mapping problem and the consequent need for a concept-based organization of factual knowledge in semantic networks. The concept of a fuzzy topic predicate is introduced and used as the basis for a concept-based topic organization which can be used to structure topically-related propositions into fuzzy categories. It is shown how these structures, when adequately defined, can provide concept-based access to topically-relevant propositions by guiding the classification and retrieval of propositions in a propositional data base (PDB). Some of the problems both presented and solved by such structures are also discussed.

Chapter 4 describes the structure and function of a propositional data base (PDB) system which provides a substrate for superimposing concept-based topic hierarchies. An overview of the PDB's structure is given before elaborating on the basic function of its classification and retrieval subsystems in terms of the PDB's focus finder, proposition matcher, and concept and topic hierarchy maintenance components. The chapter ends by presenting several examples of the operation of the PDB's classification and retrieval mechanisms.

Finally, Chapter 5 assesses the significance of the structures and ideas proposed, before presenting some ideas

for future research based on the extended representation and the concept-based topic hierarchies.

Chapter 2

Representing Knowledge with Semantic Networks

2.1 A Semantic Network Representational Schema

Many writers (e.g. Schank 1972, Anderson & Bower 1973, Schubert 1975, Woods 1975) have indicated a preference for using semantic networks to represent factual knowledge. Their most important reasons for doing so include the following:

- Semantic networks clearly indicate unique concepts and their relationships with other concepts. The one to one correspondence bewteen network concepts and network nodes provides a unique access point for knowledge about a particular concept.

- The intuitive immediacy of a semantic network structure aids in the design of computer data structures for encoding these networks. One can easily see the correspondence between concepts and unique storage locations, and between relational links and storage address pointers. Resultant structures can be

constructed in a computationally natural and elegant manner.

- The clear indication of concepts and relations between concepts suggests and aids in the development of associative-like heuristics for processing semantic networks.

The use of semantic networks to encode general factual knowledge presumes that a process for mapping factual knowledge into semantic networks is available. A graphic encoding of a hand-drawn network would be at best, clumsy. Schubert (1975) has shown how many network notations (including those of Quillian 1968, Winston 1970, Rumelhart et al. 1972, Schank 1972) can be considered to be variants of predicate calculus. Then, emphasizing this 'near-isomorphism', he clearly indicates that a predicate calculus like notation can be used for encoding network propositions in a linear form suitable for machine translation. Chapter four describes an infix style of predicate calculus which can easily be used to state facts about the world in propositional form. A program for translating this linear notation into semantic networks and maintaining a structured data base of such networks will also be described.

2.1.1 A Semantic Network Notation

The notation to be described is that of Schubert (1975). For a more detailed discussion of the notation the reader is directed to Schubert (1975) or Cercone (1975).

The most primitive unit of information to be represented is a concept node. Within a semantic network, concept nodes may represent generic concepts (e.g. DOG, CAT) or instances of generic concepts (e.g. dog11, Bruce-the-cat). An unnamed node is considered to be a variable, denoting an unspecified instance of a generic concept.

2.1.1.1 Network Propositions

The basic unit of knowledge represented by the notation is the network proposition, which will subsequently be referred to by the term 'network'. All networks consist of atomic networks and logical compounds of atomic networks. An atomic network is constructed from a proposition node and a set of concept node arguments connected to the proposition node by labelled links. One such link is a predicate link (PRED link) which impinges on the predicate (generic concept node) of the proposition. Other links (ARG links) bind individual or variable concept nodes to the proposition as required by the predicate. Figure 2.1 provides an example of an atomic network which asserts 'Fred is a dog'. Networks whose predicative concepts have more than one argument will have their ARG links labelled to indicate the

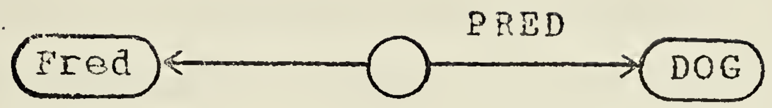


Figure 2.1 [Fred DOG]

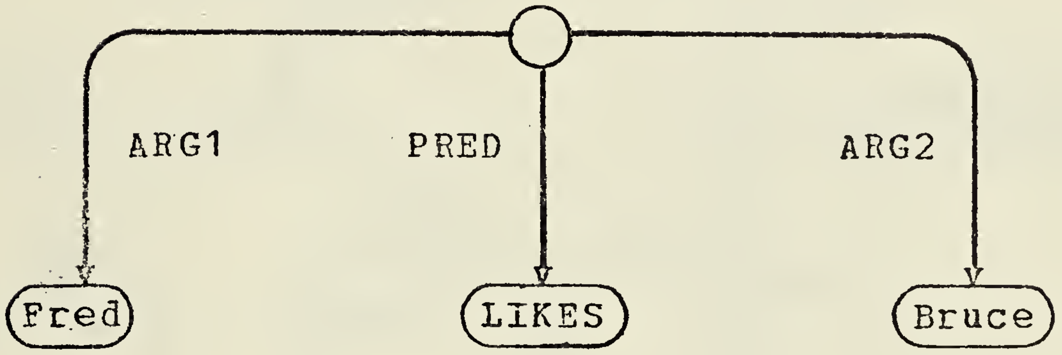


Figure 2.2 [Fred LIKES Bruce]

binding order of the argument nodes (see figure 2.2).

2.1.1.2 Network Operators and Connectives

A network operator or connective, along with a number of networks (either atomic or compound) can be linked to a proposition node with logical links (LOG links are drawn as '----->') to form a new compound network. The rules for constructing compound networks are similar to the rules for constructing a well-formed formula in predicate calculus. If 'P' is a well-formed network (wfn), then so is 'not P'

(or '¬P') and 'necessary P' (or '□P').[1] If 'P' and 'Q' are wfn's then so is 'P implies Q' (or 'P=>Q'). Conjunction and disjunction are defined to have an arbitrary number of components - if 'P', 'Q', and 'R' are wfn's so is their conjunction ('P&Q&R') or disjunction ('P|Q|R'). Figure 2.3 asserts 'If Fred is a dog and Bruce is a cat, then Fred chases Bruce'. In the previous notation of Schubert,

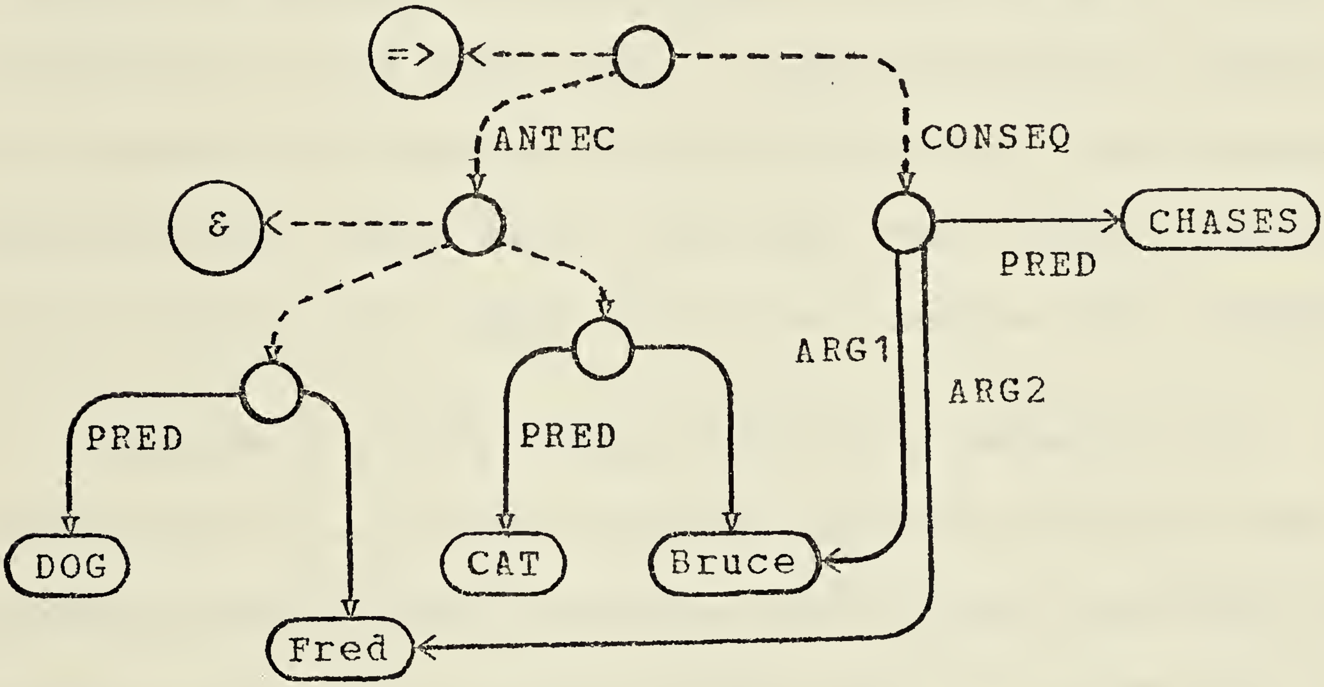


Figure 2.3 [[[Fred DOG] & [Bruce CAT]]
=> [Fred CHASES Bruce]]

implication was generalized to have an arbitrary number of antecendents and consequents, where the antecedents and consequents are regarded as conjoined propositions. However, since it may be desired to disjoin antecedents or

---

[1] The necessary operator is used to distinguish between contingent truths (e.g. crows are black) and necessary truths (e.g. crows are birds).

consequents, the binary form of implication will be retained here. Conjunction or disjunction of antecedents or consequents is shown by forming explicit conjunctive or disjunctive networks.

2.1.1.3 Quantifiers

Semantic networks have often suffered from weak or non-existent methods of quantifying variable nodes within networks.[2] It is clear that a representational schema must incorporate some method of quantification since so many propositions about world knowledge make explicit use of quantifiers (e.g. Every child has a mother and a father).

Schubert (1975) allows arbitrary embedding of quantifiers within the scope of network operators and connectives, so that networks need not be translated to a prenex form.[3] The system consists of two components, first, a method of distinguishing between existentially quantified and universally quantified nodes, and second, a method of indicating the dependencies of quantified nodes embedded within the scope of other quantified nodes and operators.

---

2. Schubert (1975) and Cercone (1975) both describe several other methods of incorporating quantifiers into semantic networks. Among those discussed are Isard & Longuet-Higgins (1971), Simmons & Bruce (1971), Palme (1971) and Anderson & Bower (1973).
3. More important than relieving the necessity of converting propositions to prenex form, allowing quantifiers to be embedded within the scope of modal operators (e.g. necessarily) preserves the distinction between transparent and opaque modal contexts (see Quine 1960 and Schubert 1975).

2.1.1.3 Quantifiers

In the graphic form, universally quantified nodes are drawn as broken circles, rendering them distinct from existentially quantified nodes which are drawn as closed circles (see figure 2.4). The scope dependencies of quantified nodes are indicated by a form of graphical 'Skolemization' using scope links (SCP links are drawn as '••••>'). For each set of *adjacent* quantifiers SCP links will be drawn from governing universally quantified nodes to each of their existential dependents. Other quantified nodes without immediately adjacent governing quantifiers have their dependence indicated by a SCP link from the proposition node of the network in which they are embedded. The resulting system of SCP links follows a path of decreasing scope from nodes of maximum scope (i.e. nodes with no impinging SCP link) to the most deeply nested nodes. Figure 2.5 shows a network proposition with with embedded quantifiers, which states 'for every human, there necessarily exists something that is the mother of that human'.

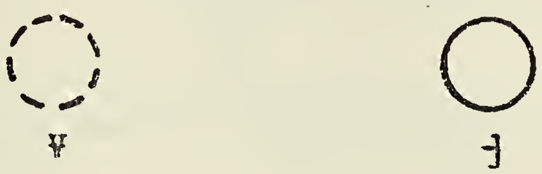


Figure 2.4 Universal and Existential Nodes

2.1.1.3 Quantifiers

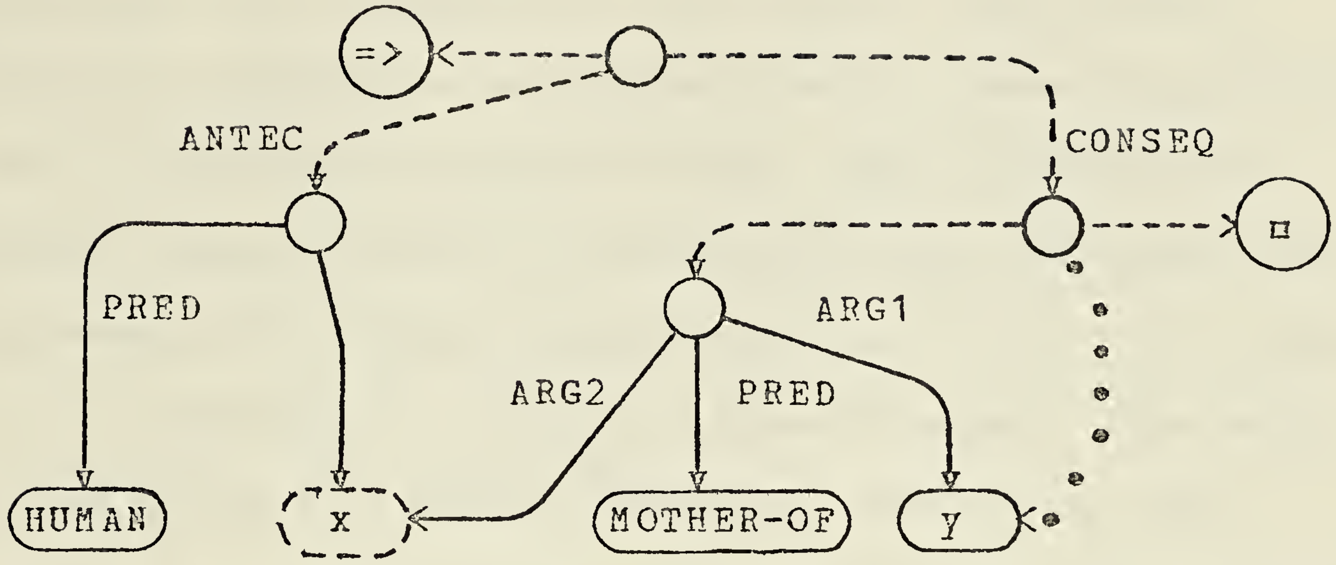


Figure 2.5 $\forall x[[x\ \text{HUMAN}] \Rightarrow [\Box\exists y[y\ \text{MOTHER-OF}\ x]]$

The representation as discussed so far is essentially equivalent (without the modal operator 'necessary') to a first-order predicate logic (cf. Rogers 1971). The interpretation intended is one where generic concepts are predicates whose extensions are subsets of the domain of individuals over which all other concept nodes representing constants and variables range.

2.1.1.4 Time

Following the example of McCarthy & Hayes (1969), Schubert assigns a central role to time by associating time arguments with all 'physical' predicates. Up to two time indicators are allowed to be bound with the use of each generic concept. These time arguments can be used to indicate the time (instant or interval) that a proposition if in effect. Propositions without attached time arguments

may be interpreted as timeless (i.e. time independent) or permanent (i.e. holding for all times). One or two arguments indicate a particular instant (one) or interval (two) a particular proposition is, was, or will be in effect. This version of the notation will indicate time arguments with TIMEARG links (drawn as '■ ■ ■ ■ >') which will be labelled to indicate argument order. Time nodes may be quantified to indicate universal moments of time. Assuming 'x' and 'y' are people, and 'z' and 'w' are moments of time, figure 2.6 states 'everybody loves somebody for some period of time'. Concept nodes representing time can be used together with 'time' predicates (e.g. BEFORE, AFTER, DURING) to assert propositions about relative moments and intervals of time (see Cercone 1975, Cercone & Schubert 1975 for a more detailed discussion).

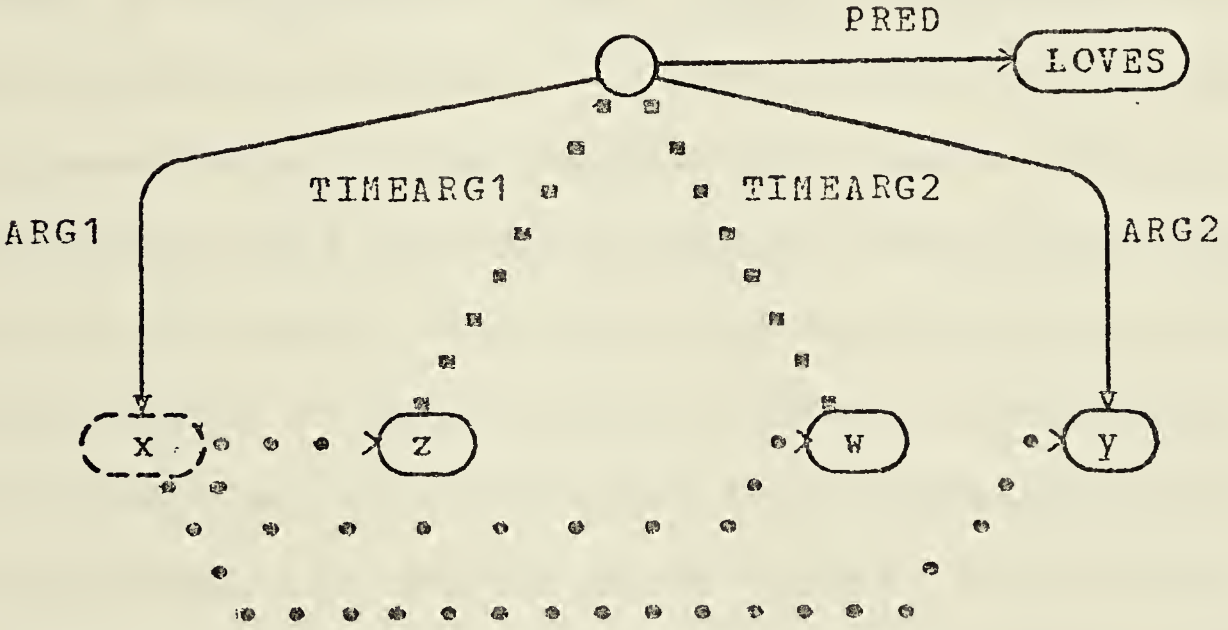


Figure 2.6 $\forall x \exists y \exists z \exists w [x \text{ LOVES}\langle z,w\rangle \, y]$

2.1.1.5 Truth and Credibility

> It does not trouble people much that their heads are full of incomplete, inconsistent, and uncertain information. With little trepidation they go about drawing rather doubtful conclusions from their tangled mass of knowledge, for the most part unaware of their reasoning. The very tenuousness of the enterprise is bound up with the power it gives people to deal with a language and a world full of ambiguity and uncertainty (Collins et al., 1975, page 383).

The vague and uncertain nature of human knowledge coupled with man's unique ability to reason with such knowledge is a complex and poorly understood facet of human intelligence. Indeed, much of man's intellectual ability might be attributed to this 'power' to operate within vague and uncertain contexts. It seems natural that a formal representation of the vagaries and uncertainties of knowledge would provide Artificial Intelligence research with a formidable tool for artificial reasoning systems. A brief review of some of the Artificial Intelligence literature which investigates the representation and understanding of belief and vagueness follows below. This discussion provides a background for a proposal which incorporates a model of vagueness and credibility into semantic networks.

In general, vagueness has been identified with degrees of truth and modelled with various formalisms based on

multi-valued logical systems, especially fuzzy logic.[4] Although Gaines & Kohout (1977) state that the general literature of fuzzy system theory has had 'little impact' on the literature of Artificial Intelligence, some work has been reported (e.g. Lee & Chang 1971, Lee 1972, Kling 1974, LeFaivre 1974, 1976, 1977) and others have admitted to the potential utility of such systems(e.g. Bobrow & Winograd 1976, Hendrix 1976) of such systems. While degrees of truth have usually been interpreted and manipulated in an ad hoc way in Artificial Intelligence, they have been the subject of formal study by logicians and mathematicians (e.g. Rosser & Turquette 1952, Ackerman 1967). This formal background has been a precursor to the development of some programmed models of fuzzy systems. For example, Kling (1974) and LeFaivre (1974) use a value on the interval [0,1] to represent the truth value of a proposition stated in a PLANNER (Kling) or LISP-like (LeFaivre) notation. Both use the standard 'min' and 'max' rules for combining truth values over conjuncts and disjuncts of propositions, and indicate two alternatives in the choice of a detachment rule for computing truth values of fuzzy inferences. That is, for two propositions 'p' and 'p=>q' we want to derive a truth value for the consequent 'q'. One alternative is that of Lee (1972) which uses

---

[4] See Gaines & Kohout (1977) for a comprehensive survey of fuzzy systems literature, as well as a short discussion of its genesis and evolution.

$$T(q) = \min\{T(p), T(p \Rightarrow q)\}.$$

Thus the truth value of a chain of implications is the truth value of the 'weakest link'. Goguen's (1968) alternative uses

$$T(q) = T(p)\ T(p \Rightarrow q),$$

consequently the truth value of a chain of implications decreases with the length of the chain.

Kling (1974) and LeFaivre (1974) both imply that the selection of a rule for computing the truth values in detachment is domain dependent. More specifically, the selection depends on the predicates and their arguments within a domain. For example, consider the predicte 'NEAR'. The assertions 'Vancouver is NEAR Victoria' and 'the moon is NEAR the earth' may be equally true, but they clearly imply different frames of reference. NEAR in this case is really a comparison with some reference distance. Successive applications of the transitivity of NEAR will result in a more or less true proposition as long as the locations of objects involved in the proposition are within the reference distance specified or implied. Once they are further apart than this reference distance, further transitive deductions are false, but neither Lee's nor Goguen's scheme will assign zero truth value if all intermediate truth values are non-zero.

Degrees of belief (i.e. credibilities), like degrees of truth, have also been used heuristically in some Artificial Intelligence systems (e.g. Schank & Rieger 1974). However, within Artificial Intelligence more attention has been devoted to investigating the representation and structure of 'belief systems' (e.g. Abelson 1973, Becker 1973, Colby 1973, Kulikowski 1974, Weiss 1974, Shortliffe & Buchanan 1975, Schmidt & Sridharan 1977a,b). The general motivation for investigating 'belief systems' is the assumption that an intelligent system must be able to refer to a model of its own beliefs and those of its conversational partners in order to exhibit intelligent behaviour in the real world.

Becker (1973), Shortliffe & Buchanan (1975), Kulikowski (1974), and Weiss (1974) all propose systems which use various probability models to represent what can be interpreted as a model of uncertain or inexact knowledge. Becker (1973) discusses a cognitive model for 'encoding experiential information' based on the application of pattern driven 'schemata', a kind of event-based implication rule. A confidence weight attached to each schema indicates the frequency of its success (how often a match of the antecedent resulted in the match of a consequent) and is used as an a priori probability to predict how successful the next application of that schema will be.

Shortliffe & Buchanan (1975) develop the notion of a 'certainty factor' (CF) as the basis for an 'inexact'

reasoning model operating within an anti-microbial therapy program (MYCIN). They argue that when a system's body of knowledge becomes large, a Bayesian probability model based on a priori evidence becomes unmanageable because of the large numbers of statistical dependencies which must be analyzed. They propose the use of a heuristic system of 'evidential strength' based on an analysis of several formal probabilistic systems (e.g. Carnap 1950, Hempel 1965). The program's knowledge is encoded in production rules (a form of implication rule) which consist of a set of antecedents (clinical facts) and a consequent or diagnostic hypothesis. To test a hypothesis, the minimum CF of its antecedents is multiplied by the CF attached to the production rule to arrive at the hypothesis CF, which is then compared with an empirically determined validation threshold.

In a program for diagnosing glaucomas (CASNET), Kulikowski (1974) and Weiss (1974) utilize a 'causal network' of pathophysiological states connected by weighted causal arcs as a basis for reasoning about more or less credible information. Each state represents a clinical fact, and has an attached strength value which is used in conjunction with an acceptability threshold to guide a causal reasoning mechanism through the associative net. This casual reasoning system determines 'admissable paths' of causation in a network of states, then each state in an admissible path receives a 'status value' determined as a function of its evidential strength and the product of the

weights on causal arcs leading to that state. This value indicates the relative degree of confirmation or denial of the hypothesis represented by that state.

The MARGIE system of Schank et al. (1975) includes what may be interpreted as a measure of credibility. Internal conceptualizations (propositions) have an attached 'STRENGTH' value on the interval [0,1] indicating the system's degree of belief of that conceptualization. The 'Conceptual Memory' component of the system uses 'inference molecule' dependent rules to arrive at STRENGTHs for inferred conceptualizations.

There is often some confusion in deciding whether a proposed system is a model of belief (i.e. subjective credibility) or a model of vagueness (i.e. conceptual fuzziness), especially in systems that model one or the other but not both. LeFaivre (1977) has shown how the essential components of the inexact reasoning systems in both CASNET and MYCIN can be modelled in his FUZZY programming language, since FUZZY simply attaches a value to a statement and allows the user to interpret the value any way he wishes. As LeFaivre says,

> The term 'Z-value' was chosen so as not to give any semantic connotations to the usage of this numeric modifier, since it can represent a conventional truth-value, a fuzzy set grade of membership, a degree of certainty or belief, a simple weight, or anything else the user wishes (LeFaivre, 1976, page 1071).

Of course attaching a single value on the interval [0,1] to a proposition can be interpreted as representing a truth

value or degree of subjective belief, but not both. For Artificial Intelligence, both are necessary since any adequate understanding system must be able to represent not only the vagueness inherent in real world concepts, but also the uncertainty of its beliefs about them.

The notation proposed here models both credibility and fuzziness by attaching a credibility distribution function over truth values to each network proposition. For example, suppose the discrete distribution over truth values in figure 2.7 was attached to the propositon

[Fred YOUNG].

The concept young can be interpreted as a distribution of of degrees of truth over ages. Therefore the credibility distribution of truth values says something about Fred's likely age. For example, since the credibility of zero truth value is 0.35, there is a 35 percent likeihood that Fred is some age which is not considered 'young' at all (say, > 40). Or, since the credibility of truth value 0.7 is 0.05, there is a 5 percent chance that Fred's degree of 'youngness' is 0.7 (say, 25 - 30). This idea seems similar to Zadeh (1977), although at the time of this writing further details of his 'PRUF' system were not available.

For pragmatic reasons, each credibility function over truth values will be represented as a 'staircase' type of discrete distribution function (i.e. constant except for a

```
  1.0 -|
      -|
  0.9 -|
      -|
C 0.8 -|
r     -|
e 0.7 -|
d     -|
i 0.6 -|
b     -|
i 0.5 -|
l     -|
i 0.4 -|
t     -|*
y 0.3 -|     *
      -|
  0.2 -|               *
      -|
  0.1 -|                         *
      -|                                   *
  0.0 -+|----|----|----|----|----|----|----|----|----*----*
       0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0

                               Truth
```

Figure 2.7 A discrete credibility distribution

finite number of upward jumps) Fp(t), with steps arbitrarily chosen at

$$t = 0.0,\ 0.1,\ 0.3,\ 0.5,\ 0.7,\ 0.9,\ 1.0$$

such that

$$Fp(t) = P(\ T(p) \leq t)$$

where Fp(t) is the probability that the truth value of proposition p, T(p), is less than or equal to t (see figure 2.8). At each step t, the probability that the truth value is ≤ t is the sum of the probabilities of the previous steps.

for t =	0.0	0.1	0.3	0.5	0.7	0.9	1.0
P(T≤t) =	0.35	0.65	0.85	0.95	1.00	1.00	1.00

```
  1.0 -|                                                  *
      -|                         *************************
  0.9 -|
      -|               **********
C 0.8 -|
r     -|
e 0.7 -|
d     -|     **********
i 0.6 -|
b     -|
i 0.5 -|
l     -|
i 0.4 -|
t     -|*****
y 0.3 -|
      -|
  0.2 -|
      -|
  0.1 -|
      -|
  0.0 -+|----|----|----|----|----|----|----|----|----|----|
       0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0

                                Truth
```

Figure 2.8 A discrete cumulative credibility distribution

Using these distributions in conjunction with reasoning processes requires rules for combining distributions over truth functions. For example, for independent propositions 'p' and 'q' the combining rules for for 'pVq', 'p&q' and '¬p' can be derived as follows:

Rule D (logical disjunction)

$$FpVq(t) = P(T(pVq) \le t) \qquad (1)$$

$$= P(\max\{T(p),T(q)\} \le t) \qquad (2)$$

$$= P(T(p) \leq t \,\&\, T(q) \leq t) \qquad (3)$$

$$= P(T(p) \leq t)\ P(T(q) \leq t \,||\, T(p) \leq t) \qquad (4)$$

Assuming 'p' and 'q' independent we have

$$FpVq(t) = P(T(p) \leq t)\ P(T(q) \leq t) \qquad (5)$$

$$= Fp(t)\ Fq(t). \qquad (6)$$

Rule C (logical conjunction)

$$Fp\&q(t) = P(T(p\&q) \leq t) \qquad (7)$$

$$= P(\min\{T(p),T(q)\} \leq t) \qquad (8)$$

$$= P(T(p) \leq t \mid T(q) \leq t) \qquad (9)$$

$$= P(T(p) \leq t) + P(T(q) \leq t)$$

$$- P(T(p) \leq t \,\&\, T(q) \leq t) \qquad (10)$$

$$= Fp(t) + Fq(t) - FpVq(t) \qquad (11)$$

again assuming 'p' and 'q' independent,

$$= Fp(t) + Fq(t) - Fp(t)\ Fq(t) \qquad (12)$$

Rule N (logical negation)

$$F\neg p(t) = P(T(\neg p) \leq t) \qquad (13)$$

$$= P(1 - T(p) \leq t) \qquad (14)$$

$$= P(T(p) \geq 1 - t) \qquad (15)$$

$$= 1 - P(T(p) < 1 - t) \qquad (16)$$

$$= 1 - P(T(p) \leq 1 - t)$$

$$+ P(T(p) = 1 - t) \qquad (17)$$

$$= 1 - Fp(1 - t) + P(T(p) = 1 - t) \qquad (18)$$

A rule for implication can be derived as

$$Fp{=}{>}q(t) = F\neg p \vee q(t). \qquad (19)$$

Rules for detachment can be derived similarly to correspond to either the definition of Goguen (1968) or Lee (1972). Examples of distributions derived from application of these rules are found in Appendix 1.

The belief in a proposition can be verbalized by examining the credibility distribution in question at a particular truth value (say 0.5) and mapping the credibility value at that point onto a set of descriptive terms similar to the set given in table 2.1 below. For example, if the cumulative credibility distribution of figure 2.8 was attached to the proposition 'it will rain tonight', when

$1.00 = Fp(0.5)$	=> 'certainly not'
$0.80 < Fp(0.5) \leq 0.99$	=> 'probably not'
$0.60 < Fp(0.5) \leq 0.80$	=> 'perhaps not'
$0.40 < Fp(0.5) \leq 0.60$	=> 'uncertain'
$0.20 < Fp(0.5) \leq 0.40$	=> 'perhaps'
$0.00 < Fp(0.5) \leq 0.20$	=> 'probably'
$0.00 = Fp(0.5)$	=> 'certainly'

Table 2.1 A Credibility Interpretation

queried 'will it rain tonight?', a system might appeal to the interpretation of table 2.1 to answer 'probably not'.

If the step function Fp(t) has reached the value 1.0 before t = 0.5, the system believes the proposition is necessarily false, that is truth values greater than or equal to 0.5 are not at all possible. Alternatively, if Fp(t) is 0.0 up to and including t = 0.5, the proposition is interpreted as being necessarily true, since it has no probability of having a truth value less that 0.5. This interpretation is useful when we want the system to accept a proposition by fiat. For example, we might require the system to accept the proposition

$$\forall x[[x \text{ DOG}] \Rightarrow [x \text{ MAMMAL}]]$$

as true by definition. In this way, meaning postulates and other necessary truths can be installed by hand, without fear of their being interpreted as falsifiable.

Chapter 3

Organizing Knowledge in Semantic Networks

> Retrieval is very much a matter of retrieving enough contextual information to delimit a small enough space of memory to carry out some sort of memory search...we get a good deal of savings in search time by indexing information in terms of the context in which we store the information - or, better yet, in terms of the context in which we expect to have to retrieve the information (Rumelhart, 1976, page 358).

3.1 Introduction

The term 'knowledge systems' within Artificial Intelligence generally refers to theories that attempt to impart the ability of 'knowing' to a machine. Assuming that knowledge and the use of knowledge is a vital component of human intelligence, it is natural to desire an analogous component for machines. Recent research in Artificial Intelligence has stressed the development of organizational theories of knowledge such as Scripts (Abelson 1973, Schank 1975c, Schank & Abelson 1975), Frames, (Minsky 1974), Schemata, (Bobrow & Norman 1975). This trend more or less succeeds a previous one that produced various representational schemata for machine knowledge like the

Semantic Networks of Quillian (1968), the PROGRAMMAR Procedures of Winograd (1972), the Predicate Calculus System of Sandewall (1971a), and the Production Systems of Newell & Simon (1972). Many of these representations have provided a facility for representing only relatively simple units of knowledge like a natural language sentence or a basic rule of inference. Thus the current interest in extending the representations with superimposed organizations follows from a desire to store and retrieve increasingly complex knowledge structures. The relative size and complexity of the knowledge domains in existing knowledge-based systems (e.g. Carbonell 1970, Woods et al. 1972, Bobrow et al. 1977) imply that more sophisticated structures must be developed before an artificial system can function usefully in a real world domain. In general, solutions for organizing large and complex knowledge domains emphasize schemes which make each item of knowledge easily accessible in the contexts or situations in which it is needed. This implies the need for a context-sensitive mechanism for classifying a body of knowledge.

3.2 Semantic Network Organizations

Many writers (e.g. Quillian 1968, Schank 1972, Anderson & Bower 1973, Norman & Rumelhart 1975) have shown how semantic networks can provide an elegant representation for knowledge, but recently, the organizational capabilities of such networks have been questioned. Bobrow (1975) indicates

that "Predicate calculus and semantic network representations tend to impose only a local organization on the world," which might imply that such notations are not suitable for representing higher level organizational structures. Ironically, although many of the most popular knowledge organization structures are described as being built of such things as declarative descriptions (e.g. frames, schemata) or episodes (e.g. scripts, plans), they can ultimately be viewed as a kind of network, as indicated by Kay (1976) speaking on the frame-driven dialogue system GUS:

> ...now the contents of these slots in the dialogue frames (and in lots of other frames that exist in the system) are typically other frames. These structures recurse to great depth. Of course they are not simply tree structures, but they are circular and they point to one another; they're networks (Kay, 1976, page 355).

It seems then, that some of the 'flavour' of semantic networks is inherent in these larger units of knowledge, although many of them are not explicitly constructed with a network notation.

One higher level structure explicitly developed with a form of semantic network is the 'script' of Abelson and Schank. The base representation for scripts is Schank's (1972) Conceptual Dependency (CD) notation, a type of semantic network. CD is used to build 'conceptualizations', which become the basic units for constructing 'actions' or 'episodes'. These episodes are combined in a cause-effect structure to represent a "predetermined, stereotyped

sequence of actions" called a script. Scripts are intended to be a specialized frame (cf. Minsky 1974) useful for understanding episodic knowledge like "going to a restaurant". They are not intended as a tool for organizing knowledge in general.

Much of the research directly involved with predicate calculus and semantic network type notations has been done within the context of natural language understanding (e.g. Sandewall 1971, Palme 1971, Schank 1972, Simmons 1973, Cercone 1975). This work has emphasized the construction of network expressions to represent isolated natural language concepts and utterances, with less emphasis on developing techniques for organizing these structures into larger units of knowledge. The relatively fewer examples of research aimed specifically at the 'bottom-up' development of organizations for semantic networks have emphasized the organization of represented concepts. For example, Winograd (1975) discusses a 'generalization' hierarchy which can be used for organizing concepts in a structured knowledge model. The structure is essentially a sub-concept/super-concept hierarchy in which the concept nodes are viewed as 'frames'. The 'frames' are used as attachment points for knowledge about that concept, in fact each 'frame' is actually a collection of facts describing the concept being represented. Mylopoulos et al. (1975) develop an organization of semantic networks in the 'frame' tradition, calling their corresponding structure a 'scenario'.

Scenarios are composed of 'concepts', 'characteristics' (properties of concepts), and 'events' organized by causal and temporal connectives to form representations of stereotypic knowledge. This knowledge is organized around two basic structural components, namely the 'IS-A' hierarchy, and the 'PART-OF' hierarchy.

Other efforts to impose organizations on sematic networks have been made in order to extend their expressive power. For instance, Hendrix (1975) extends a semantic network by incorporating a notion of partitioning in order to represent quantification and modal contexts. These constructions are already handled by the notation discussed in Chapter 2.

Many of these recent proposals for extending the organizational power of semantic networks provide methods for assembling propositions into larger units, but little effort is devoted to organizing the collections of facts associated with each concept in the network (i.e. those facts which can be said to be 'about' the concepts). The next portion of this chapter is directed at developing such an organization.

3.3 The Symbol-Mapping Problem

The 'Symbol-Mapping problem' is actually a spate of various problems that arise in the following situation described by Fahlman (1975):

> Suppose I tell you that a certain animal-- let's call him Clyde-- is an elephant. You accept this simple assertion and file it away with no apparent display of mental effort. And yet, as a result of this simple transaction, you suddenly appear to know a great deal about Clyde. If I say that Clyde climbs trees or plays the piano or lives in a teacup, you will immediately begin to doubt my credibility. Somehow, "elephant" is serving as more than a mere label here; it is, in some sense, a whole package of properties and relationships, and that package can be delivered by means of a single IS-A statement (Fahlman, 1975, page 7).

From an Artificial Intelligence point of view, the problem is one of organizing a system's factual knowledge so that when it is told 'Clyde is an elephant', every other fact known about elephants becomes immediately accessible to it from the concept node 'Clyde'. Fahlman's solution involves the use of parallel hardware elements to represent concepts (e.g. Elephant) and relations (e.g. IS-A) between concepts. Each unit can store 'marker-bits' which can be propagated in parallel through a network of such elements in order to very quickly perform searches and intersections on large concept and relation classes. This provides the efficient access necessary to solve the basic problem with a minimum of organizational overhead.

Serial approaches emphasize various organizations of conceptual properties, and generally attempt to provide

efficient concept-based access of known properties. McDermott (1975a) suggests organizing properties of concepts into 'packets' or 'contexts' similar to the devices available in PLANNER, CONNIVER or QA4. When an instance of a concept is instantiated (e.g. when 'Clyde the elephant' is asserted) the corresponding packet of properties, in this case the 'elephant packet', is attached to Clyde by some form of variable binding or indexing. This makes all known elephant properties accessible from the concept Clyde.

Moore's (1975) scheme focusses on the use of an 'IS-A' taxonomy to organize the facts associated with particular concepts (see figure 3.1). For example, since ELEPHANT is

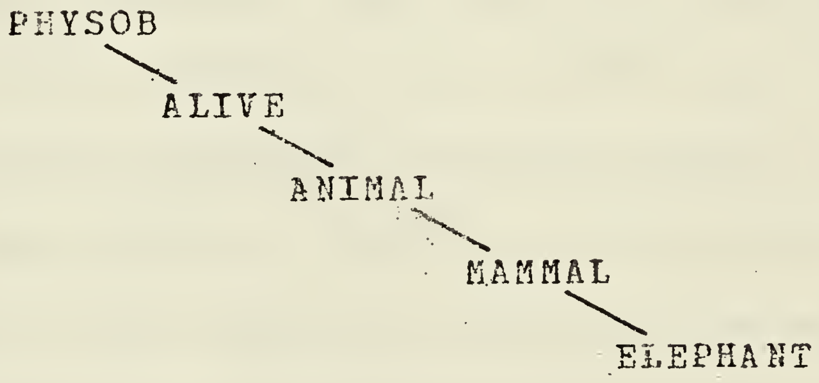


Figure 3.1 Part of Moore's IS-A taxonomy

classified under MAMMAL, asserting 'CLYDE IS-A ELEPHANT' implies an inheritance of known properties from all the other nodes in the IS-A taxonomy. Efficient access of inherited properties is facilitated by attaching a list of subsuming class concepts to each constant (i.e. extensional object) or variable (i.e. intensional object) about which

properties have been asserted. For example

(OCCUPY-SPACE ?x/PHSYOB)

asserts that all physical objects occupy space. Asserting

(CLASS CLYDE ELEPHANT)

attaches the class list 'PHYSOB ALIVE ANIMAL MAMMAL ELEPHANT' to 'CLYDE', and the task of verifying the fact

(OCCUPY-SPACE CLYDE/PHYSOB ALIVE ANIMAL MAMMAL ELEPHANT)

is left to a pattern matcher. The rules for matching variables (e.g. '?x'), constants (e.g. CLYDE) and class lists determine which inherited properties can be verified by the matcher, thus making it the key to the whole scheme's success. The key to the matcher's efficient operation pivots on how quickly it can find the assertions it is looking for, therefore Moore proposes that assertions be grouped in hierarchical buckets by class lists, then the class lists can be used to index the desired assertion buckets. When a bucket becomes to large to search efficiently, it is subdivided into more manageable sub-buckets as specified by the next sub-concept attached to the assertions of the large bucket. For example, subdividing the bucket for assertions about the 'PHYSOB' class might result in sub-buckets for the assertions about the 'PHYSOBJ

ALIVE' class, and the 'PHYSOB NOT-ALIVE' class. This subdividing process can carry on down the class hierarchy to only the depth of constants, at which point the organization would resemble a more or less balanced 'IS-A' hierarchy.

3.3.1 Symbol-Mapping in a Semantic Network

Symbol-Mapping in a semantic network is greatly aided by imposing a sub-concept super-concept (IS-A) hierarchy on the network concepts.[1] Within the semantic network, one concept is a <u>super-concept</u> of another concept if the set of properties attached to the former is a <u>subset</u> of the properties attached to the latter. Therefore 'MAMMAL' is a super-concept of 'ELEPHANT' since the set of 'MAMMAL' properties is a subset of the set of 'ELEPHANT' properties.

Now, after asserting 'Clyde is an elephant', the colour of Clyde can be searched for as follows: The concept node 'Clyde' is accessed, and all attached propositions are scanned sequentially for one which indicates the colour of Clyde. Should such a property not be found for Clyde, the class concept 'ELEPHANT' is accessed, and its attached properties are again searched sequentially for a colour

---

[1] The sub-concept/super-concept structure is actually part of the semantic network, although it can be discussed as being distinct from it. In general, a sub-concept/super-concept relation between generic concepts can be asserted as '∀x[[x P] => [x Q]]', or 'P' is a sub-concept of 'Q'. Asserting that '[x P]' does not mean that 'x' is a sub-concept of 'P', but that 'x' is an instance of the class 'P', and that 'x' inherits the properties of 'P' and its super-concepts.

proposition. This process continues until either all the existing super-concepts' properties have been checked, or a colour proposition has been found. Notice that the sub-concept super-concept (S/S-CONCEPT) structure simply directs an exhaustive search for a colour property attached to each of the super-concept classes of which Clyde is a member, but it does little to increase the efficiency of locating relevent information about the colour of Clyde. What we would like is to be able to ask the question 'is there a colour proposition attached to Clyde?', and if the answer is no, proceed up the S/S-CONCEPT hierarchy, asking the same question of each successive super-concept.

3.3.2 Topic Predicates

Topic predicates will provide a tool for classifying atomic network propositions into similar categories or topics. Topic predicates will take propositions as arguments to form topic propositions whose truth value represents the degree to which the propositon argument belongs to that topic. Topic assertions are triggered when a topically classified predicate is recognized in an input proposition. The topic classification is based on the definition of sub-topic/super-topic relations between topic nodes, for example table 3.1 defines the 'texture' topic to consist of the sub-topics 'smooth', 'bumpy', 'ragged' and 'shiny'. Each topic node (those appearing in lower-case type) has a corresponding first-order predicate (appearing

```
{smooth SUBTOP texture}
{bumpy SUBTOP texture}
{ragged SUBTOP texture}
{shiny SUBTOP texture}
```

Table 3.1 The 'texture' topic

in upper-case type) which is included by definition in that corresponding topic. Therefore 'texture' topic assertions can be triggered by any of the first-order predicates 'BUMPY', 'RAGGED', 'SHINY', or 'TEXTURE'. Attaching the first-order 'trigger' predicates in this way keeps the topic relations consistent, since only topic predicates may participate in a 'SUBTOP' relation.

The sub-topic lists for two different topics need not be disjoint. For instance, the topic 'colour', like the topic 'texture' may also have a sub-topic 'shiny', since shiny objects tend to have ill-defined colours (e.g. consider a mirror), so 'shininess' substitutes for colour (see table 3.2). When the corresponding first-order predicate 'SHINY' is used in a proposition, that proposition is classified under the 'texture' topic and the 'colour' topic.

Topic propositions formed with a proposition containing a topic's corresponding first-order predicate are assigned unit credibility. For example a 'smooth' topic proposition whose proposition argument uses the first-order predicate

{red SUBTOP colour}
{green SUBTOP colour}
{blue SUBTOP colour}
•
•
•
{shiny SUBTOP colour}

Table 3.2 A partial definition for the 'colour' topic

'SMOOTH' would have a truth distribution function $Fp(1.0) = 1.0$ and $Fp(t) = 0.0$ for all $t < 1.0$. Since propositions classified under a topic are also included in that topic's super-topic categories, it is more interesting to consider the degree to which a topic proposition satisfies it super-topic classification. The topic defining statements are themselves propositions, therefore topic categories can be 'fuzzified' by attaching a truth distribution to each sub-topic (or super-topic) definition.[2] When a proposition is categorized by a topic predicate, the attached truth distribution gives the degree to which the classified proposition satisfies its topic category. As previously stated, if the truth distribution function $Fp(t)$ is 0.0 up

---

[2] Note that the concept of a fuzzy topic predicate is similar to the concept of a fuzzy set (cf. Zadeh 1965). Note also, that topic predicates resemble 'linguistic variables' conceived by Zadeh (1974), although the interpretation intended is not linguistic, but epistemic. LeFaivre (1977) shows how fuzzy sets and linguistic variables can be represented in his programming language FUZZY, and his examples indicate that fuzzy topic predicates could as well be accomodated.

to, and including t = 0.5, a proposition is interpreted as being necessarily true. The definitions of the two fuzzy topic predicates in table 3.3 have all the truth values for their distribtuion functions accumulated at one of the values t = 0.7, 0.9, 1.0. In a propositional data base,

$$Fp(t)=1.0 \ \& \ Fp(x)=0.0 \ \forall x < t \Rightarrow P(T(p)=t)=1.0$$

{red SUBTOP colour} Fp(1.0) = 1.0
{green SUBTOP colour} Fp(1.0) = 1.0
{blue SUBTOP colour} Fp(1.0) = 1.0
•
•
•
{dark SUBTOP colour} Fp(0.9) = 1.0
{shiny SUBTOP colour} Fp(0.7) = 1.0

{smooth SUBTOP texture} Fp(1.0) = 1.0
{bumpy SUBTOP texture} Fp(1.0) = 1.0
{ragged SUBTOP texture} Fp(0.9) = 1.0
{shiny SUBTOP texture} Fp(0.9) = 1.0

Table 3.3 Two fuzzy topic predicates

classifying asserted propositions with the fuzzy topic predicates 'texture' and 'colour' not only provides two viewpoints of the concept 'SHINY', but also indicates which of the asserted propositions is the best representative of each topic category. If all that has been asserted about an object is that it is 'DARK' and 'SHINY', an accessing mechanism can immediately infer that 'DARK' is a better indicator of colour than 'SHINY'. In general, overlap in the definition of many such topics can provide alternate

[illegible]

[illegible]

[illegible]

viewpoints of the same concept to varying degrees of relevance as indicated by the credibility distributions of the topic predicate definitions.

3.3.3 Topic Hierarchies

The sub-topic/super-topic (S/S-TOPIC) relations can be used to define arbitrary hierarchies of topics which correspond to heuristic classifications of propositional knowledge. Such hierarchies imply an inheritance of categorized knowledge similar to the inheritance of properties implied by a sub-concept/super-concept (IS-A) organization. The definition of the 'appearance' topic in table 3.4 implies that items of knowledge classified under the 'colour' topic are also items of knowledge about 'appearance'. This enables a partially-filled structure (which will usually be the case) to provide access to classified knowledge from any topic level - only a simple inference must be made to access propositions from any one

$$Fp(t)=1.0 \;\&\; Fp(x)=0.0 \;\forall x < t \Rightarrow P(T(p)=t)=1.0$$

{structure SUBTOP appearance} $Fp(1.0) = 1.0$
{colour SUBTOP appearance} $Fp(1.0) = 1.0$
{texture SUBTOP appearance} $Fp(0.9) = 1.0$
{pattern SUBTOP appearance} $Fp(0.9) = 1.0$

Table 3.4 The 'appearance' topic

of a topic's sub-topic or super-topic categories.

The purpose of defining topic hierarchies is to impose a classification on a body of propositions in order to provide efficient access of topically-relevant propositions for a given query. This efficiency depends on how well a topic hierarchy 'covers' the domain of knowledge being classified. To keep the access time of propositons within a reasonable range (e.g. approximately constant with respect to the number of propositions classified), the topic hierarchy must be somewhat balanced; it should have approximately the same number of propositions under each category. Attempting to maintain such a balance will usually result in topics at any level of the structure being of different levels of generality. For example, consider the partial topic hierarchy for organizing factual knowledge about physical objects given in figure 3.2. In this hierarchy both 'appearance' and 'behaviour' appear as topic predicates at the same level, but the sub-topics attached to the 'appearance' topic will classify propositions to a greater depth than those attached to the 'behaviour' topic. A similar notion appears in the Symbol-Mapping problem solution proposed by Moore (1973). Assertions about concepts are stored in assertion buckets, and when a bucket becomes too large to search 'efficicently' it is sub-divided into more manageable sub-buckets. Therefore the sub-bucket structure attached to each concept may not always have the same depth.

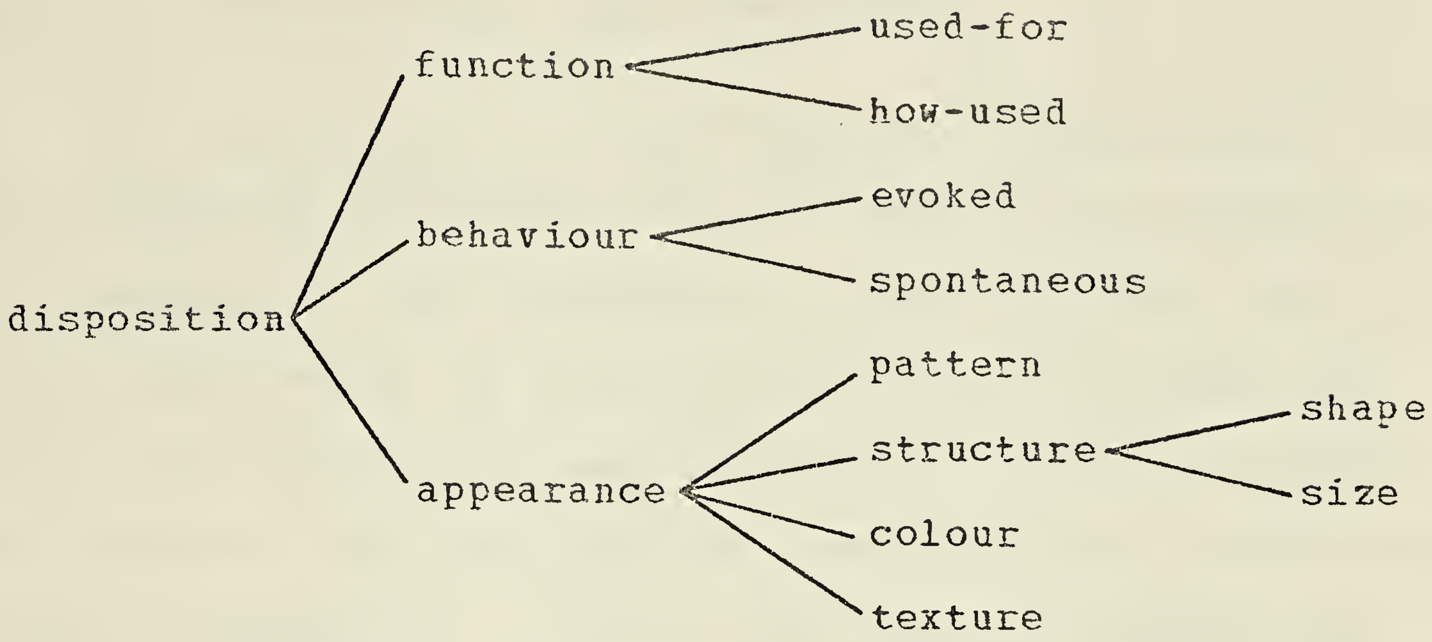


Figure 3.2 A partial topic hierarchy

3.3.3.1 Concept-based Topic Hierarchies

Now reconsider the Symbol-Mapping problem assuming that the propositions attached to each concept are classified by an appropriate topic hierarchy. A search for the colour of Clyde again begins by accessing the 'Clyde' concept, but rather than looking at each proposition attached to Clyde, the 'colour' topic of Clyde can immediately be checked for a colour proposition. If one is not found, the super-topic structure of the generic classes in which Clyde is a member (e.g. ELEPHANT) are used to direct a similar search of topically classified knowedge. If the topic hierarchies attached to each concept are roughly balanced, the access time for a particular classified proposition about a particular concept will be approximately proportional to the logarithm of the number of propositions 'known' about that concept. The combined organizational power of the S/S-CONCEPT and S/S-TOPIC structure should provide for a significant reduction in proposition access time.

Having topic hierarchies attached to each semantic network concept requires that the system determine which concepts a proposition is 'about' before it can be classified under those concepts' topic structures. The problem is one of finding the <u>concept</u> <u>foci</u> of a proposition. The notion of a concept focus is similar to the 'topic concept' notion of Charniak (1972, 1973, 1976), which he uses as an access and attachment point for knowledge about that concept. Pattern-driven 'demons' (i.e. antecedent

theorems in PLANNER) are attached to each topic concept and invoked by 'base-routines' when the concept is recognized in an asserted proposition. The demons then make logical deductions about the concept within the current context as inferred from the asserted proposition. Topic hierarchies provide a similar access and attachment point for knowledge, and since they are concept-based, the concept foci of a proposition must be determined before that propostion can be correctly classified.

There are basically two types of concept foci that can appear in a proposition; individual concept foci, and generic concept foci. This distinction is made by determining how predicative arguments in a proposition are quantified. Individual concept arguments are recognized as unquantified appearances of named arguments denoting instances of generic concepts (e.g. Fred, ball17), or existentially quantified arguments denoting an unspecified individual. For example, in the proposition

$$\exists y[\text{Fred LIKES } y]$$

both 'Fred' and 'y' denote individual nodes in the semantic network. Since they are the concept foci, the proposition will be classified under the topic hierarchies of both 'Fred', and a newly created, unnamed node representing some individual.

The most informative cue for recognizing the generic

foci of a proposition is the presence of universally quantified arguments denoting sets of individuals in the domain. For instance, the proposition

$$\forall x[[x\ \text{BEAR}]\ \&\ [x\ \text{WHITE}]] \Rightarrow [x\ \text{POLAR-BEAR}]]$$

has only one unique argument, and since it is universally quantified, the proposition must have generic foci. The proposition is about all 'BEAR' things, all 'WHITE' things, and all 'POLAR-BEAR' things, therefore it will be classified under the topic hierarchy of each of those generic concepts. It may seem somewhat counter-intuitive to classify the proposition as knowledge about 'WHITE', perhaps because of the different nature of that adjectival concept as compared to the noun concepts 'BEAR' and 'POLAR-BEAR'. One seldom considers knowledge about 'WHITE' things since they generally have little in common. Conversely, noun concepts like 'BEAR' and 'POLAR-BEAR' imply a larger set of consequences (e.g. they are mammals, have claws, etc.), so the proposition seems potentially more valuable for recognizing a 'BEAR' or a 'POLAR-BEAR', but relatively less valuable for recognizing a 'WHITE' thing. However, this does not provide a basis for deciding that the proposition is not knowledge about the concept 'WHITE'.

There are cases when a concept may have both individual and generic foci. For example, the proposition

$$\forall x[[x\ \mathrm{HUMAN}] \Rightarrow [\mathrm{Fred\ LIKES}\ x]]$$

surely says something about 'Fred', but also makes reference to the set of all 'HUMAN' things and the set of all things which are the 'LIKES' of 'Fred'. A general rule for determining the foci in such situations would be as follows: first determine how each argument is quantified; any individuals are concept foci. The appearance of at least one universally quantified argument indicates the presence of generic foci; the generic foci will be exactly those predicates which have a universally quantified argument. By this rule, the concept foci of the above proposition are 'Fred', 'HUMAN', and 'LIKES'.

Chapter 4

Utilizing Knowledge in Semantic Networks

4.1 Introduction

Demonstrating the use of topic hierarchies requires both a propositional data base (PDB) for constructing and maintaining a body of propositions, and a computational model of the topic hierarchy structure which can be integrated with the PDB system. This chapter describes such a PDB system, and illustrates the design and operation of a topic organization which can be used to classify and retrieve propositions stored in the PDB.

4.1.1 Efficiency of Retrieval

Efficiency of retrieval will ultimately depend on how many propositions must be examined before the 'right one' is found. If the number of propositions is very small, the organizational overhead involved in maintaining a topic structure might result in a loss of efficiency, but then it is doubtful that such a small body of knowledge would be very useful in a real world environment.

Ideally, the hierarchical organization should have its categorized propositions distributed more or less evenly in order to maintain the same number of propositions under each topic. Maintaining a nearly balanced structure will keep the access time for a proposition approximately logarithmic with respect to the number of propositions classified. In a static topic hierarchy, the balance of the structure is determined by the distribution of predicates and arguments appearing in the input propositions. Whether or not the structure remains balanced will depend on how well the predefined topic structure 'covers' the anticipated body of input propositions. This suggests that a self-adaptive structure would be valuable, although the current model makes no attempt to investigate that possibility.

4.2 The PDB Framework

A PDB system has been programmed to provide a facility for constructing and maintaining a semantic network data base which, in turn, will provide a substructure upon which an S/S-TOPIC structure can be superimposed. The description of the PDB system which follows clearly demonstrates how an actual implementation could be programmed. The status of the currently implemented prototype system is discussed in Appendix 3, and some actual 'dialogue' with the system is given.

Before dicussing examples which demonstrate the use of the organizing devices, a brief description of the PDB is

necessary. The description is intended only to familiarize the reader with the basic PDB components and their interaction with the organizational structures.

4.2.1 The PDB Network Structure

Each node in the semantic network is represented by a unique record consisting of various fields depending on the type of node (proposition, predicate, constant, or variable) being represented. All links to other nodes are represented by a LINK record consisting of type and pointer fields.

4.2.1.1 Propositions

Each proposition node is represented by a PROP record consisting of five basic fields, four of which are list pointers. The single scalar (atomic) field is the assertion time of the proposition as given by the PDB's internal clock. This field imposes a chronological ordering on asserted propositions, and can be used as an aid to various kinds of inferring techniques (e.g. see Schank & Rieger 1974). The propositional argument list (PARGLIST) is a list of LINK records pointing to the arguments of the proposition. For each atomic proposition, the PARGLIST consists of a PRED type LINK to a generic concept and a set of ordered ARG type LINKs to the arguments of the predicate. The PARGLIST of a compound proposition has a LOGICAL type LINK to the appropriate operator ('¬' or '¤') or connective ('=>', '|', or '&') followed by LOGICAL type LINKs to the

various constituent propositions. For implications, subcompounds are ordered antecedent, then consequent. Lists of disjuncts and conjuncts are sorted by the index of the first predicate node (i.e. generic concept) found in each subjunct. This helps the system to determine if two differently ordered disjunctions or conjunctions are equivalent. For example, if the predicates 'DARK' and 'HORSE' appear in the internal dictionary in that order, the asserted conjunct

[[Jack HORSE] & [Jack DARK]]

will be stored as

[[Jack DARK] & [Jack HORSE]]

The scope inclusion list (SCPLIST) is a (possibly empty) list of SCOPE type LINKs pointing to quantified nodes included with the scope of the proposition (as described in Chapter 2, Section 2.2.1.3). The remaining list field is the back LINK list (PBAKLIST) which is made up of LINK records pointing from sub-compound PROP nodes to their parent proposition nodes. This list can be used to determine if a proposition is in someway modified by another, for example a parent proposition that negates the subproposition, or indicates that it is a consequent of another proposition, etc..

The credibility distribution list (CREDLIST) is a list

of numeric values defining the credibility and truth of the proposition. The current version of the PDB system maintains a seven-valued discrete distribution as described in Chapter 2, Section 2.1.1.5.

4.2.1.2 Generic Concepts

All generic concept nodes (i.e. those that can act as predicates) are represented by CONCEPT records. Each CONCEPT record consists of four fields, two of which are lists. The atomic fields include a pointer to the concept's dictionary entry, and a count of the number of times the concept has been used as a predicate. The predicate use count (CPREDCNT) provides a simple heuristic for deciding with which of several generic classes a search for a multi-class referent should begin. For example, a search for the referent 'x' in the proposition

[[x BALL] & [x BLUE]]

might begin by determining whether each object in the BLUE class is also in the BALL class, or vice versa. If the CPREDCNTs indicate there are more BLUE objects than BALL objects, the search should naturally begin by examining each object in the BALL class (i.e. because there are fewer of them) and determine if any of them are BLUE.

The predicate usage list (CPREDLIST) consists of LINKs pointing to each proposition in which an individual concept

appears as a predicate. This list delimits the set of individual concepts included in the class defined by the generic concept. Its function may be pre-empted by a special topic in a PDB which uses a superimposed topic organization.

In a PDB model without a superimposed topic structure, the CDEFLIST provides an attachment point for propositional knowledge about the proper use of the generic concept as a predicate. For example, the fact that the relation concept NEAR is transitive could be represented by the proposition

$$\forall x \forall y \forall z[[[x \text{ NEAR } y] \ \& \ [y \text{ NEAR } x]] \Rightarrow [x \text{ NEAR } x]].$$

Since this information is in semantic network form, it can be interpreted with the same tools used to interpret all other knowedge stored in the PDB. This means a general reasoning mechanism can be used to reason about predicate usage. In a PDB system with an integrated topic organization, the propositions attached to the CDEFLIST may be classified under a 'special' topic for such knowledge. This notion will be elaborated later.

4.2.1.3 Individual and Variable Concepts

Both individual and variable concepts are represented by NODE records consisting of three fields, one dealing with the quantification apparatus, another with dictionary access, and the last with the NODE record's usage as an

argument in propositions.

Individual concepts (e.g. Fred, ball7) have dictionary links and a NULL scope list (NSCPLIST) indicating that they are existentially quantified instances of generic concepts which can be referred to by name. Variable nodes do not have dictionary entries and can be referred to only by their participation in the propositions in which they were specified. The NSCPLIST of existentially quantified variables is NULL, while universally quantified variables have a (possibly empty) NSCPLIST consisting of SCP type LINKS to any existentially dependent variable nodes. Both variable and constant NODE records have an attached list of LINKs to propositions in which they appear as arguments (NARGLIST). This list delimits the propositional knowledge the system 'knows' about a particular concept at any given moment. Once again, a PDB system with an embedded topic organization may choose to retain the NARGLIST entries for propositions which cannot be topically-classified, or alternatively, may classify them under a special 'miscellaneous' topic.

4.2.2 The PDB Organizational Structure

A topic hierarchy may be superimposed on the PDB by organizing each concept's back links into separate categories or lists, one for each topic containing classified propositions in which the concept appears as a concept focus. It is unlikely that any topic classification

scheme will be complete, and propositions which simply cannot be classified by topic can either be added to a miscellaneous topic, or made accessible through the CPREDLISTs and NARGLISTs of the individual concepts. In systems using several different kinds of topic hierarchies (e.g. one for knowledge about physical objects, one for knowledge about actions, one for knowledge about events, etc.) there may be considerable overlap in the lists representing the topic classifications since topic predicates' sub-topic lists need not be mutually exclusive.

The topic organization has two components, a predefined topic hierarchy (e.g. the topic hierarchy in Chapter 3, figure 3.2), and a topic access skeleton attached to each CONCEPT or NODE record under which propositions have been classified.

The topic hierarchy is represented by a set of TOPIC node records whose sub-topic and super-topic relations are represented by SUBTOP and SUPTOP links. Each TOPIC node consists of a dictionary index and two (possibly empty) lists for the SUBTOP and SUPTOP links. Each link contains a pointer to a TOPIC node, and a pointer to a credibility distribution which represents the truth distibution of the defined relation. Each generic concept (i.e. first-order predicate) which has a corresponding topic predicate will have a 'trigger' link pointing to the topic node in question. A topic node can then be accessed via its predicate 'trigger' or by its dictionary entry, then the

topic hierarchy can be traversed by following that topic's list of connecting links.

Each node in a topic access skeleton is a list of propositions which have been classified by the corresponding topic node in the topic hierarchy. Assertion or retrieval of a proposition is done by classifying the proposition in the topic hierarchy, then performing the appropriate operation (i.e. insertion or retrieval) on the list located at the corresponding topic access skeleton node.

Each topic access skeleton is represented as a linked list structure 'isomorphic' to the topic hierarchy, and therefore need not be fully specified for topics under which propositions have not been classified. Representing partial rather than complete topic access skeletons saves storage, but still allows access to propositions through the appropriate sub-topics. When a proposition is classified under a topic for which a focus concept's access skeleton does not have an entry, the complete branch of the skeleton's new topic will be instantiated. For example, if all that is 'known' about the concept 'Fred' is that it is spotted, the topic access skeleton for 'Fred' would be something like figure 4.1.

Notice that in figure 4.1, the proposition about 'Fred' being 'SPOTTED' has not been classified under the corresponding 'spotted' topic category. In general, if a topic predicate has no sub-topics, it can be deleted from

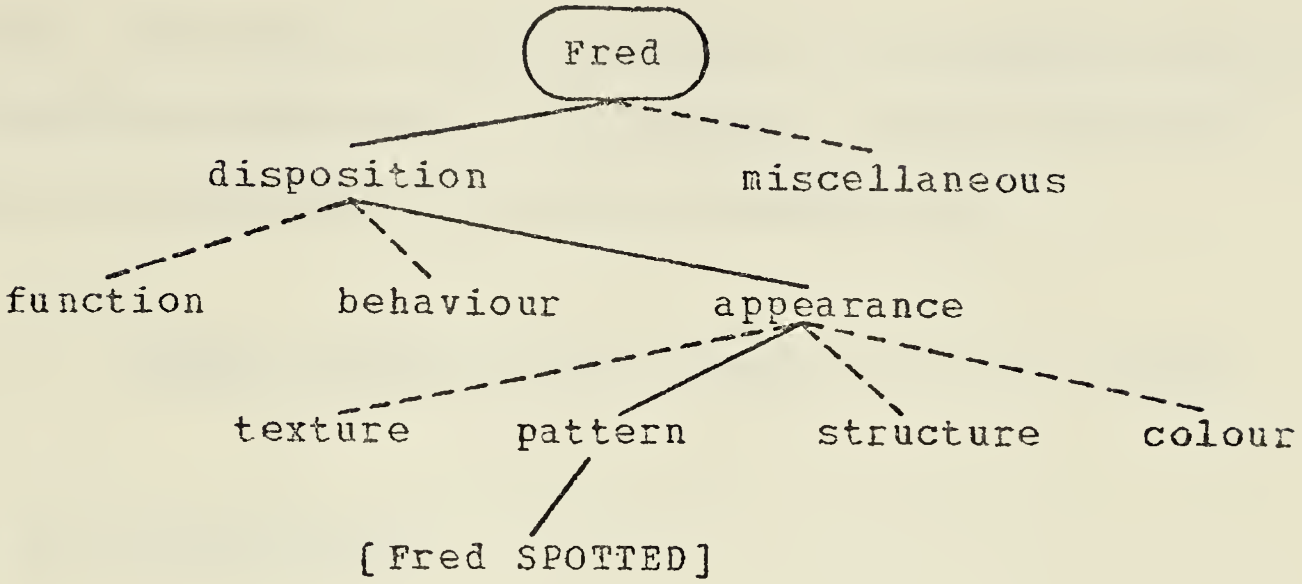


Figure 4.1 A partial topic access skeleton for 'Fred'

the topic access skeleton since it will have at most one entry. It can still be accessed in the topic hierarchy, and only need appear in the topic access skeleton if a sub-topic is added to it.

The topic hierarchy can be made to accommodate 'special' topics which will categorize other kinds of knowledge about concepts. For example, as previously mentioned, the CDEFLIST of concept properties could be made a special topic in order to make concept usage information generally accessible. A 'miscellaneous' topic for attaching propositions wich cannot be classified would replace the function of the CPREDLIST on predicate nodes, and the NARGLIST on individual nodes.

Topic access to S/S-CONCEPT information can be facilitated by adding 'sub-concept' and 'super-concept' topics to the topic structure. A super-concept (or sub-concept) predicate can be defined in terms of a semantic

network, then can be used to trigger the classification of S/S-CONCEPT assertions. For example, if the following proposition defines a predicate 'SUBCON' as

$$\forall P \forall Q[[P \text{ SUBCON } Q] <=> [\Box \forall x[[x\ P] => [x\ Q]]]],$$

then the proposition

[DOG SUBCON MAMMAL]

would be classified under the 'sub-concept' topic of 'MAMMAL', and under the super-concept of 'DOG'. Note that if necessary, the predicate 'SUBCON' can be interpreted within the logic of the semantic network since the connective '<=>' could be defined as a combination of implications.

4.2.3 PDB Utilization

The two major functional components of the PDB are the classification system and the retrieval system. Since the two are difficult to disengage, their functions will be discussed in terms of the utility sub-systems which they employ. The description of these utility components will be somewhat stylized since the system does not always easily decompose into separate units. Identifying the separate modules is more a matter of recognizing the current function of the operational system rather than isolating distinct

structural components. This is more a consequence of the level of description being attempted rather than of a non-modular approach to the programming of the PDB.

4.2.3.1 Encoding Propositions

Propositions are entered in an 'English-like' infix form of predicate calculus which is transformed into the internal semantic network representation described in Chapter 2.

Currently, when a proposition is entered the concept nodes in the proposition are created if they cannot be found in the system's internal dictionary. An unquantified argument is assumed to be the name of an individual concept, and if it does not exist, it is created as specified. All quantified arguments are entered as new variables of the appropriate type (i.e. existential or universal). Predicates may be optionally entered as recognized on input, or predefined by hand and verified upon input. The input syntax includes a functional notation which permits reference to an individual concept by its participation in an atomic network proposition. For instance, the functional expression

(FATHER-OF Fred)

is a functional reference to the FATHER-OF Fred, whoever he may be. The system currently allows functional reference to

existing nodes, but will not create new nodes which are implicitly defined. In general, the decision to create a new node will require a more cautious approach, since indiscriminant creation of new nodes could quickly lead to data base inconsistency. For example, if the proposition

[Sally SISTER-OF Fred]

has been asserted, the the functional references

(FATHER-OF Sally)
(FATHER-OF Fred)

refer to the same father, but reasoning is required to recognize this fact in order to refrain from creating two new NODE records.

Credibility distributions are specified by including a 'T*' within a proposition, following the specified arguments, viz.

[Fred CHASES Bruce T*].

The parser removes the 'T*' from the proposition, and prompts the user for the appropriate values defining the cumulative credibility distribution as described in Chapter 2, Section 2.1.1.5.

The input syntax permits compounds of arbitrary number

of disjuncts or conjuncts (e.g. A & B & C &...), and any proposition may appear as an argument, for example

[Fred BELIEVES [Bruce CAT]].

Propositions can be assigned external labels on input which can be used to refer to that proposition for the remainder of the input session. For example, the proposition

[Fred DOG =P1]

may be subsequently referred to as ':P1', and used in further inputs, viz.

[Bruce KNOWS :P1].

This syntax is merely for convenience, since it avoids reentering lengthy statements.

All predicate calculus statements appearing in the text of this thesis are examples of propositions which the PDB's parser can transform into the semantic network notation. A complete description of the syntax for constructing propositional forms is given in Appendix 2.

4.2.3.2 The Classification and Retrieval Systems

After the PDB parser has correctly parsed an input, either the classification or retrieval system takes control. The PDB classification and retrieval components can be thought of as goal-oriented control structures, differing only in the actions they take upon achieving (as is possible) their primary goals. From this viewpoint, the primary goal of each system might be described as isolating the location of a prescribed propositional form in the PDB semantic network. Having accomplished this, the respective system proceeds either to insert new information in the network, or to extract old information situated at the isolated location. The ultimate recourse for the retrieval system is failure; after all the inferring methods for interpreting the prescribed propositional form have been unsuccessful in locating relevant information, the retrieval system must resign. Alternatively, classification never fails completely since propositions will always be made accessible through their constituents or a 'miscellaneous' topic even if they cannot be topically-classified.

4.2.3.3 Utility Components

There are four more or less distinct components which the classification and retrieval control mechanisms can utilize to manipulate a propositional form. These include the focus finder, the proposition matcher, and the S/S-TOPIC and S/S-CONCEPT maintenance routines. Since the decision to

use any of these utilities rests with the supervising control structure, the relative level of reasoning sophistication achieved is largely dependent on the decision making machinery of the controlling system. Within the current version of the PDB, no attempt has been made to build a more globally motivated control structure which could oversee the decisions made by the classification and retrieval subsystems.

The focus finder's purpose is to receive a proposition and construct an ordered list of its concept foci. In doing so, the focus finder may call upon the services of the proposition matcher or the S/S-CONCEPT and S/S-TOPIC routines. The built-in plan for finding foci begins at the atomic proposition level, where the quantification of arguments gives a first indication of whether the proposition has generic or individual focus. Atomic propositions with individual focus can immediately be passed to the S/S-TOPIC maintenance routines to initiate classification. Propositions with possible generic foci must be further processed according to the rules given in Chapter 3, Section 3.3.3.1.

The basic function of the S/S-CONCEPT and S/S-TOPIC maintenance routines is to verify queries about the existing relationships between topic or concept nodes, and to make additions and deletions to each structure as appropriate. For example, the focus finder might query the S/S-CONCEPT structure to determine if 'BEAR' is a subconcept of

'MAMMAL', or the retrieval system could interrogate the S/S-TOPIC hierarchy for the super-topics of the 'colour' topic.

The proposition matcher provides a basic pattern-matching utility to be used as required by the classification and retrieval control systems. For classification, the matcher is a convenient (and necessary) tool for comparing predicates during classification, and for confirming the equivalence of propositions. In retrieval, the matcher is used to facilitate pattern-directed access of propositions. Only syntactic matching is done, but universally and existentially quantified variables can be used to match sets and individual nodes respectively. Compound propositional forms are matched by decomposing the compound into atomic components, then recomposing each matched subproposition until a mismatch is encountered or the proposition has been fully reconstructed. The matcher can be instructed to retain the partial results of such a reconstruction match for possible interpretation by other components of the system. This is potentially useful for pattern-directed back tracking, an issue not further pursued in the current system.

4.2.3.4 Examples

The operation of the PDB classfication and retrieval systems can be made clearer with a few examples. Consider the following assertion:

```
[[Bruce CAT =P0 ] & [Bruce FURRY =P1 ] =P2 ].
```

The PDB parser constructs temporary parser tables representing the three propositions P0, P1, P2 and each of their predicates and arguments. The assert module begins by attempting to associate all the specified concepts with existing concepts in the semantic network. As mentioned above, the system creates new nodes for arguments only if they cannot be located in the internal dictionary. Once the locations of all predicates and arguments have been found, the propositions are <u>inserted</u> into the network, beginning with the most deeply nested. In this case the order of insertion will be P0, P1, and P2. As a simple method of improving the efficiency of search, the syntactic form of each proposition is used to derive an index to a proposition reference vector through which propositions can be retrieved by their form. For example, the two forms

```
[[   ] & [   ] & ...   ]
[[   ] => [   ]]
```

will be stored in different index 'buckets'. This indexing scheme is a simple version of the co-ordinate indexing used in PLANNER (Hewitt 1971) and QA4 (Rulifson et al. 1972).

Note that although the propositions have been inserted into the PDB network, they have not yet been asserted. At this point the inserted proposition P2 can only be accessed

by sequentially scanning the proposition bucket indexed by the form

[[ ] & [ ]].

If compound constituents of propositions where asserted as they were inserted, logical contradictions could arise, as would happen if both disjuncts in the propositon

[[Fred DOG] OR [¬[Fred DOG]]

were asserted. In general, propositions are not considered to be asserted until they have been classified by the topic hierarchy mechanism. The classification procedure begins by checking to see if the proposition can be split into separately classifiable sub-propositions. This is determined on the basis of quantifier scopes, and the types of logical operators and connectives. In this case the proposition is made up of two distinct conjuncts, so they may be classified separately. The focus finder indicates that the concept focus of both propositions P0 and P1 is 'Bruce', so an attempt is made to classify the two propositions under the topic access skeleton of the 'Bruce' concept. Using the topic hierarchy for physical objects given in Chapter 3, figure 3.2, P0 cannot be classified, but P1 can be inserted under the 'disposition - appearance - texture - furry' branch of Bruce's topic access skeleton. The proposition P0 is assigned to a 'miscellaneous' topic

(see figure 4.2).

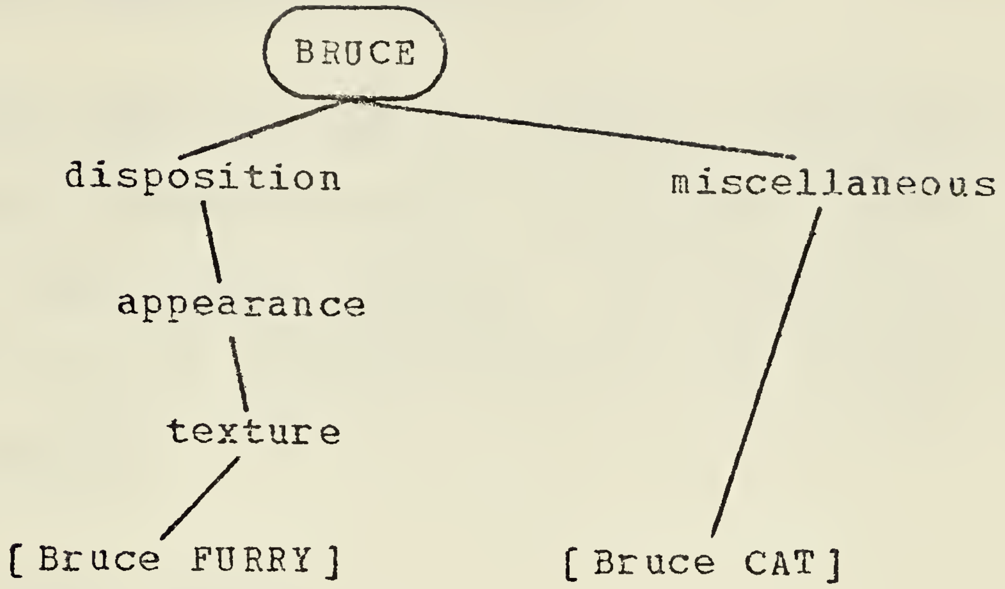


Figure 4.2 A partial topic access skeleton for 'Bruce'

When the parser recognizes a query, the retrieval system is given control immediately after the parser tables have been constructed and before any insertions are done. Three basic types of query are recognized, the functional query, the topic query, and the propositional query.

A functional query is simply passed to the function evaluator to attempt evaluation. For example, to evaluate the function

(FATHER-OF Fred)?

the function evaluator attempts to match the proposition

∃x[x FATHER-OF Fred]

and if successful, returns the name of the concept node bound to the variable concept 'x'.

A topic query is used to specify a retrieval of the propositions classified under a particular topic in a concept's topic access skeleton. The syntax for initiating a retrieval of all propositions about the 'colour' of 'Clyde' would be

Clyde; colour

The retrieval process is as follows: After the parser recognizes the statement as a topic query, a procedure travels down the SUBTOPIC links from the 'colour' topic of Clyde's topic access skeleton adding categorized propositions to a list of relevant propositions.

The last type of query which can initiate a retrieval process is a more general form which is specified as an arbitrary propositional form with an appended '?'. The retrieval system searches the PDB and attempts to find valid bindings for any quantified variables specified in the query. As an example, consider the process initiated by the proposition

$\exists y[[y \text{ TAIL}] \& [y \text{ PART-OF Fred}]]?$

Upon recognizing the proposition as a query, the parser passes control to the query routine. The focus finder is

used to determine that the foci of this proposition are the individual concept 'Fred' the existentially quantified variable 'y'. Because the statement is a query, the system is looking for a valid binding for 'y', and since 'Fred' is the only other concept focus, the logical place to search first is 'Fred''s topic structure. Neither sub-compound contains a topic predicate, so the 'miscellaneous' topic must be scanned for a proposition which matches

$$\exists y[y \text{ PART-OF Fred}].$$

If a match is found, the 'y' is bound, and the proposition

$$[y \text{ TAIL}]$$

must be found before the query can be answered (in this instance the bound value of the variable 'y' is used).

If this process fails, the process is repeated for the concepts in which 'Fred' is a member. These concepts are found by scanning the 'miscellaneous' topic of 'Fred' for unary predicates indicating that 'Fred' is a member of a particular class of concepts. If the proposition can be matched with any of the propositions attached to any of these class concepts or their super-concepts, the system can deduce the proposition

$$\exists y[[y \text{ TAIL}] \,\&\, [y \text{ PART-OF Fred}]].$$

If each super-concept has been unsuccessfully searched, the query cannot be answered.

Chapter 5

CONCLUSIONS

> ...the body fills us with loves and desires and fears and all sorts of fancies and a great deal of nonsense, with the result that we literally never get an opportunity to think at all about anything (Plato).[1]

5.1 Significance

Topic hierarchies and credibility distributions are the two most salient features of this research. It has been shown how a topic hierarchy structure can be imposed on a data base of propositions in order to provide access to topically-relevant propositions about a particular concept. Combined with a sub-concept super-concept structure, the topic hierarchy provides a serial (as opposed to parallel) solution for the Symbol-Mapping problem.

It was argued that representations for the vagueness of concepts and uncertainty of propositions must be included in a serious knowledge representation, and a distribution of

---

1. in Tredennick, H. (trans.), The Last Days of Socrates, Peguin Books, Harmondsworth, England, page 111.

truth values was proposed to represent those notions. It was indicated how a system could verbalize its credibility distributions to express an appropriate degree of skepticism about what it knows, and how credibility distributions could be combined over logical compounds of independent propositions. Credibility distributions were combined with topic predicates to form fuzzy topic predicates, which subsequently provided a facility for the fuzzy classification of propositions.

The PDB system demonstrates the 'near-isomorphism' between predicate calculus and semantic networks. It provides a general facility for constructing a data base of propositions in semantic network form, upon which various organizational structures can be superimposed and demonstrated. Although the topic organization proposed was defined as a distinct structure superimposed upon the PDB, it could be defined within the semantic network notation. This would allow the system to interpret its own organizational knowledge, and potentially lead to adaptive self-modification. When this is done, the distinction between <u>organized</u> and <u>organizing</u> knowledge can be made only on the basis of how each bit of knowledge is used. Unless the system has use of its organizing knowledge, that knowledge cannot be considered 'meta-knowledge' from the system's point of view.

5.2 Plans for future research

Topic hierarchies were developed as a solution to the Symbol-Mapping problem, but their use is not specifically restricted to organizing factual knowledge about physical objects. In general, they can be used to define any type of heuristic knowledge classification and therefore could provide organizations for knowledge about other kinds of concepts like events, actions, etc.. Experiments with other types of knowledge taxonomies (e.g. Charniak 1975a) would naturally lead to investigations of multi-taxonomic organizations and of the interaction between classification hierarchies, since an intelligent system will need to organize more than just knowledge about physical objects. For example, it will have to 'know about' actions, events, and abstract objects (e.g. numbers), and topic hierarcies for organizing that knowledge will be necessary. This declarative type knowledge is only one aspect, since the complementary procedural knowledge about how to use physical objects, perform actions, develop and use plans, etc., will also benefit from a topic organization. The interaction and use of the declarative and procedural topic hierarchies should provide a fruitful avenue of research.

Investigating the possibility of self-adaptive organizations is another major area of research. In particular, a self-organizing strategy wuld attempt to maintain balanced topic hierarchies by splitting oversized categories and combining undersized ones. This would

require the development of an adaptive clustering algorithm of some type which would form topic clusters as a function of some identifiable feature or set of features in the input stream (e.g. co-occourence of predicates). Research and results from the linguistic information retrieval literature (e.g. Sparck-Jones & Kay 1973) are relevant to this pursuit. Resulting topic hierarchies capable of adapting to a particular environment would also reduce the effort necessary to define context dependent organizations by hand.

The development of planning systems and control structures is another vital area of research. The semantic network representation presented has much more expressive power than the current PDB can intelligently utilize (e.g credibility distributions) simply because its control structure is minimal. Steps toward developing a more powerful control system include the construction of a general purpose fuzzy reasoning system. Ideally, the system should have a globally-motivated, context-sensitive planning component which would oversee the operation of subordinate components, including the usage of the topic hierarchies.

References

Abelson, R. (1973): "The Structure of Belief Systems", *Computer Models of Thought and Language*, Schank, R. & Colby, K. (eds.), W. H. Freeman, San Francisco, 287-339.

Abelson, R. (1975): "Concepts for Representing Mundane Reality in Plans", *Representation and Understanding*, Bobrow, D. & Collins, A. (eds.), Academic Press, New York, 273-309.

Ackerman, R. (1967): *Introduction to Many-Valued Logics*, Routledge & Kegan Paul, London, England.

Anderson, J. & Bower, G. (1973): *Human Associative Memory*, V. H. Winston & Sons, Washington, D. C.

Becker, J. (1973): "A Model for the Encoding of Experiential Information", *Computer Models of Thought and Language*, Schank, R. & Colby, K. (eds.), 397-434.

Bobrow, D. (1968): "Natural Language Input for a Computer Problem-Solver", *Semantic Information Processing*, Minsky, M. (ed.), MIT Press, Cambridge, Massachusetts, 146-226.

Bobrow, D. (1975): "Dimensions of Representation", *Representation and Understanding*, Bobrow, D. & Collins, A. (eds.), Academic Press, New York, 1-34.

Bobrow, D. & Collins, A. (eds.) (1975): *Representation and Understanding*, Academic Press, New York.

Bobrow, D., Kaplan, R., Kay, M., Norman, A., Thompson, H. & Winograd, T. (1977): "GUS, A Frame-Driven Dialog System", *Artificial Intelligence 8*, 155-173.

Bobrow, D. & Norman, A. (1975): "Some Principles of Memory Schemata", *Representation and Understanding*, Bobrow, D. & Collins, A. (eds.), Academic Press, New York, 131-149.

Bobrow, D. & Raphael, B. (1974): "New Programming Languages for Artificial Intelligence Research", ACM Computing Surveys 6, 155-174.

Bobrow, D. & Winograd, T. (1976): "An Overview of KRL, A Knowledge Representation Language", Research Report, Xerox PARC, Palo Alto, California, May.

Carbonell, J. (1970): "Mixed-initiative man-computer instructional dialogues", Unpublished Doctoral Dissertation, MIT.

Carnap, R. (1950): Logical Foundations of Probability, University of Chicago Press, Chicago, Illinois.

Cercone, N. (1975): "Representing Natural Language in Extended Semantic Networks", Technical Report TR75-11, Dept. of Comp. Sci., U. of Alberta, Edmonton, Canada, August.

Cercone, N. & Schubert, L. (1975): "Toward a State Based Conceptual Representation", Proc. IJCAI 4, Tblisi, Russia, August, 83-90.

Charniak, E. (1972): "Toward a Model of Children's Story Comprehension", AI Lab Memo AI-TR-266, MIT.

Charniak, E. (1973): "Jack & Janet in Search of a Theory of Knowlege", Proc. IJCAI 3, Stanford, California, August, 337-343.

Charniak, E. (1975a): "A Partial Taxonomy of Knowledge about Actions", Working Paper 13, Instituto per gli Studi Semantici e Cognitivi, Castagnola, Switzerland.

Charniak, E. (1975b): "Organization and Inference in a Frame-like System of Common Sense Knowledge", Working Paper 14, Instituto per gli Studi Semantici e Cognitivi, Castagnola, Switzerland.

Charniak, E. (1976): "Inference and Knowledge Part 2", Computational Semantics, Charniak, E. & Wilks, Y. (eds.), North-Holland, Amsterdam, 129-154.

Charniak, E. & Wilks, Y. (eds.) (1976): Computational Semantics, North-Holland, Amsterdam.

Church, A. (1951): "The Need for Abstract Entities in Semantic Analysis", Proc. American Academy of Arts and Sciences 80, 100-112.

Colby, K. (1973): "Simulation of Belief Systems", Computer Models of Thought and Language, Schank, R. & Colby, K. (eds.), W. H. Freeman, San Francisco, 251-286.

Collins, A. & Quillian, M. R. (1969): "Retrieval Time from Semantic Memory", J. Verbal Learning and Verbal Behavior 8, 240-247.

Collins, A. & Quillian, M. R. (1970a): "Facilitating Retrieval from Semantic Memory: The Effect of Repeating Part of an Inference", Acta Psychologica 33, 304-314.

Collins, A. & Quillian, M. R. (1970b): "Does Category Size Affect Categorization Time?", J. Verbal Learning and Verbal Behavior 9, 432-438.

Collins, A. & Quillian, M. R. (1972a): "How to Make A Language User", Organization of Memory, Tulving, E. & Donaldson, W. (eds.), Academic Press, New York, 310-349.

Collins, A. & Quillian, M. R. (1972b): "Experiments on Semantic Memory and Language Comprehension", Cognition in Learning and Memory, Gregg, L. W. (ed.), John Wiley & Sons, New York, 117-137.

Collins, A., Warnock, E. H., Aiello, N. & Miller, M. L. (1975): "Reasoning from Incomplete Knowledge", Representing and Understanding, Bobrow, D. & Collins, A. (eds.), Academic Press, New York, 383-415.

Cros, R. C., Gardin, J. C. & Levy, F. (1964): Syntol-Syntagmatic Organization Language, Gauthier-Villas, Paris, France.

Davis, R., Buchanan, B. & Shortliffe, E. (1977): "Production Rules as a Representation for a Knowledge-Based Consultation Program", Artificial Intelligence 8, 15-45.

Eaton, R. M. (1931): General Logic, Charles Scribner's Sons, New York.

Elithorn, A. & Jones, D. (eds.) (1973): Artificial and Human Thinking, Proc. NATO Symposium on Human Thinking, St. Maximin, August, 1971, Jossey-Bass, San Francisco.

Falhman, S. (1975): "A System for Representing and Using Real-World Knowledge", AI Lab Memo 331, MIT, May.

Findler, N. & Chen, D. (1971): "On the problems of time, retrieval of temporal relations, causality, and co-existence", Proc. IJCAI 2, London, England, September, 531-545.

Gaines, B. & Kohout, L. (1977): "The Fuzzy Decade: A Bibliography of Fuzzy Systems and Closely Related Topics", Int. J. Man-Machine Studies 9, 1-68.

Goguen, J. (1968): "The Logic of Inexact Concepts", Synthese 19, 325-373.

Goguen, J. (1974): "Concept representation in natural and artificial languages: axioms, extensions, and applications for fuzzy sets", Int. J. Man-Machine Studies 6, 513-561.

Gray, C. (1976): "ALAI: A Language for Artificial Intelligence", M.Sc. Thesis, Dept. of Comp. Sci., U. of Alberta, Edmonton, Canada, April.

Grofman, B. & Hyman, G. (1973): "Probability and Logic in Belief Systems", Theory and Decision 4, 179-195.

Hempel, C. G. (1965): "Studies in the Logic of Confirmation", Aspects of Scientific Explanation and Other Essays in the Philosophy of Science, The Free Press, New York, 3-51.

Hendrix, G. (1975): "Expanding the Utility of Semantic Networks through Partitioning", Proc. IJCAI 4, Tblisi, Russia, August, 115-121.

Hendrix, G. (1976): "Knowledge Representation, Manipulation and Retrieval in "Intelligent Systems"", Proc. Symposium on Advanced Memory Concepts, SRI, Menlo Park, California, June, microfiche 5, 302-328.

Hewitt, C. (1971): "Procedural Embedding of Knowledge in PLANNER", Proc. IJCAI 2, London, England, September, 167-182.

Hintikka, J. & Suppes, P. (eds.) (1970): Information and Inference, D. Reidel, Holland.

Hughes, G. E. & Cresswell, M. J. (1972): An Introduction to Modal Logic, University Paperback, Norwich, Great Britain.

Isard, S. & Longuet-Higgins, H. (1971): "Question Answering in English", Machine Intelligence 6, Meltzer, B. & Michie, D. (eds.), American Elsevier, New York, 243-254.

Kay, M. (1976): "Xerox's GUS (Genial Understander System)", Proc. Symposium on Advanced Memory Concepts, SRI, Menlo Park, California, June, mocrofiche 5, 351-360.

Kintsch, W. (1972): "Notes on the Structure of Semantic Memory", Organization of Memory, Tulving, E. & Donaldson, W. (eds.), Academic Press, New York, 249-309.

Kling, R. (1973): "Fuzzy Planner", Technical Report 168, Comp. Sci. Dept., U. of Wisconsin, Madison, Wisconsin, February.

Kling, R. (1974): "Fuzzy-PLANNER: Reasoning with inexact concepts in a procedural problem-solving language", J. Cybernetics 4, 105-122.

Kuipers, B. (1975): "A Frame for Frames: Representing Knowledge for Recognition", Representation and Understanding, Bobrow, D. & Collins, A. (eds.), Academic Press, New York, 151-184.

Kulikowski, C. (1974): "A System for Computer-Based Medical Consultation", Proc. National Computer Conference, Chicago, May.

Lee, R. (1972): "Fuzzy Logic and the Resolution Principle", J. ACM 19, 109-119.

Lee, R. & Chang, C. (1971): "Some Properties of Fuzzy Logic", Information and Control 19, 417-431.

LeFaivre, R. (1974): "The Representation of Fuzzy Knowledge", J. Cybernetics 4, 57-66.

LeFaivre, R. (1975): "The Representation of Fuzzy Knowledge", Technical Report DSC-TR-33, Comp. Sci. Dept., Rutgers U., New Brunswick, New Jersey.

LeFaivre, R. (1976): "Procedural Representation in a fuzzy problem-solving system", Proc. National Computer Conference, New York, June, 1069-1074.

LeFaivre, R. (1977): "Fuzzy Representation and Approximate Reasoning", Technical Report CBM-TR-78, Comp. Sci. Dept., Rutgers U., New Brunswick, New Jersey, March.

Lindsay, R. K. (1963): "Inferential Memory as a Basis of Machines Which Understand Natural Language", Computers and Thought, Feigenbaum, E. & Feldman, J. (eds.), McGraw-Hill, New York, 217-236.

Lindsay, R. K. (1973): "In Defense of Ad Hoc Systems", Computer Models of Thought and Language, Schank, R. & Colby, K. (eds.), W. H. Freeman, San Francisco, 372-395.

Linsky, L. (ed.) (1971): Reference and Modality, Oxford University Press, London, England.

McCarthy, J. & Hayes, P. (1969): "Some Philosophical Problems from the standpoint of Artificial Intelligence", Machine Intelligence 4, Meltzer, B. & Michie, D. (eds.), American Elsevier, New York, 463-502.

McDermott, D. (1974): "Assimilation of New Information by a Natural Language Understanding System", AI Lab Memo AI-TR-291, MIT, February.

McDermott, D. (1975a): "A Packet-Based Approach to the Symbol-mapping Problem", ACM SIGART Newsletter 53, August, 6-7.

McDermott, D. (1975b): "Very Large Planner-type Data Bases", AI Lab Memo 339, MIT, September.

Meltzer, B. (1973): "The Programming of Deduction and Induction", Artificial and Human Thinking, Elithorn, A. & Jones, D. (eds.), Jossey-Bass, San Francisco, 19-33.

Mendelson, E. (1963): Introduction to Mathematical Logic, D. Van Nostrand, Princeton, New Jersey.

Minsky, M. (ed.) (1968): Semantic Information Retrieval, MIT Press, Cambridge, Massachusetts.

Minsky, M. (1974): "A Framework for Representing Knowledge", AI Lab Memo 306, MIT, June.

Moore, R. C. (1975): "A Serial Scheme for the Inheritance of Properties", ACM SIGART Newsletter 53, August, 8-9.

Mylopoulos, J., Cohen, P., Borgida, A. & Sugar, L. (1975): "Semantic Networks and the Generation of Context", Proc. IJCAI 4, Tblisi, Russia, August, 134-142.

Norman, D. & Rumelhart, D. (1975): Explorations in Cognition , W.H. Freeman, San Francisco.

Palme, J. (1971): "Making Computers Understand Natural Language", Artificial Intelligence and Heuristic Programming, Findler, N. & Meltzer, B. (eds.), American Elsevier, New York, 199-244.

Palme, J. (1975): "The SQAP Data Base for Natural Language Information", Amer. J. Computational Linguistics, microfiche 24, 1-78.

Quillian, M. R. (1968): "Semantic Memory", Semantic Information Processing, Minsky, M. (ed.), MIT Press, Cambridge, Massachusetts, 227-270.

Quillian, M. R. (1969): "The Teachable Language Comprehender", Comm. ACM 12, 459-476.

Quine, W. V. O. (1958): "Speaking of Objects", Proc. and Addresses of the American Philosophical Association, 1957-58, Antioch Press, Yellow Springs, Ohio, 5-22.

Quine, W. V. O. (1960): Word and Object, MIT Press, Cambridge, Massachusetts.

Quine, W. V. O. (1964): "Meaning and Translation", The Structure of Language, Fodor, J. & Katz, J. (eds.), Prentice Hall, Englewood Cliffs, New Jersey, 460-478.

Quine, W. V. O. (1971): "Reference and Modality", Reference and Modality, Linsky, L. (ed.), Oxford University Press, London, England, 17-34.

Rieger, C. (1975): "Conceptual Memory and Inference", Conceptual Information Processing, Schank, R. (ed.), North-Holland, Amsterdam, 157-268.

Rieger, C. (1976): "An Organization of Knowledge for Problem Solving and Language Comprehension", Artificial Intelligence 7, 89-127.

Rogers, R. (1971): Mathematical Logic and Formalized Theories, North-Holland, New York.

Rosser, J. & Turquette, A. (1952): Many-Valued Logics, North-Holland, Amsterdam, The Netherlands.

Rulifson, J. F., Derksen, J. & Waldinger, R. J. (1972): "QA4: A Procedural Calculus for Intuitive Reasoning", AI Technical Note 73, Comp. Sci. Dept., Stanford U., Stanford, California.

Rumelhart, D. (1976): "Comments on Human Memory - Some Possible Implications for Artificial Memories", Proc. Symposium on Advanced Memory Concepts, SRI, Menlo Park, California, June, microfiche 5, 361-368.

Rumelhart, D., Lindsay, F. & Norman, D. (1972): "A Process Model of Long Term Memory", Organization of Memory, Tulving, E. & Donaldson, W. (eds.), Academic Press, New York, 198-245.

Rumelhart, D. & Norman, D. (1973): "Active Semantic Networks as a Model of Human Memory", Proc. IJCAI 3, Stanford, California, August, 450-457.

Russell, B. (1948): Human Knowledge: Its Scope and Limits, Simon & Schuster, New York.

Sandewall, E. (1969): "A Set-Oriented Property Representation for Binary Relations", Machince Intelligence 5, Meltzer, B. & Michie, D. (eds.), American Elsevier, New York, 237-252.

Sandewall, E. (1971a): "Representing Natural Language Information in Predicate Calculus", Machine Intelligence 6, Meltzer, B. & Michie, D. (eds.), American Elsevier, New York, 255-277.

Sandewall, E. (1971b): "A Programming Tool for Management of a Predicate Calculus Oriented Data Base", Proc. IJCAI 2, London, England, September, 159-166.

Sandewall, E. (1973): "Conversion of Predicate Calculus Axioms, viewed as nondeterministic programs, to corresponding deterministic programs", Proc. IJCAI 3, Stanford, California, August, 230-234.

Schank, R. C. (1971): "Intention, Memory and Computer Understanding", AI Memo AIM-140, Comp. Sci. Dept., Stanford U., Stanford, California.

Schank, R. C. (1972): "Conceptual Dependency: A Theory of Natural Language Understanding", Cognitive Psychology 3, 552-631.

Schank, R. C. (1975a): "The Role of Memory in Language Processing", Technical report, Dept. of Comp. Sci., Yale University, New Haven, Connecticut.

Schank, R. C. (ed.) (1975b): Conceptual Information Processing, North-Holland, Amsterdam.

Schank, R. C. (1975c): "The Structure of Episodes in Memory", Representation and Understanding, Bobrow, D. & Collins, A. (eds.), Academic Press, New York, 237-272.

Schank, R. C. & Abelson, R. (1975): "Scripts, Plans, and Knowledge", Proc. IJCAI 4, Tblisi, Russia, August, 151-157.

Schank, R. C. & Colby, K. (eds.) (1973): Computer Models of Thought and Language, W. H. Freeman, San Francisco.

Schank, R. C. & Rieger, C. (1974): "Inference and the Computer Understanding of Natural Language", Artificial Intelligence 5, 373-412.

Schmidt, C. & Sridharan, N. (1977a): "Knowledge-directed Inference in BELIEVER", Technical Report CBM-TR-75, Comp. Sci. Dept., Rutgers U., New Brunswick, New Jersey, January.

Schimdt, C. & Sridharan, N. (1977b): "The Plan Recognition Problem: A Hypothisize and Revise Paradigm", Technical Report CBM-TR-77, Comp. Sci. Dept., Rutgers U., New Brunswick, New Jersey, March.

Schubert, L. (1975): "Extending the Expressive Power of Semantic Networks", Proc. IJCAI 4, Tblisi, Russia, August, 158-164 (also Artificial Intelligence 7, 163-198).

Shortliffe, E. & Buchanan, B. (1973): "A Model of Inexact Reasoning in Medicine", Mathematical Biosciences 23, 351-379.

Simmons, R. F. (1973): "Semantic Networks: Their Computation and Use for Understanding English Sentences", Computer Models of Thought and Language, Schank, R. & Colby, K. (eds.), W. H. Freeman, San Francisco, 63-113.

Snyder, D. P. (1971): Modal Logic and its Applications, Van Nostrand Reinhold, New York.

Spark-Jones, K. & Kay, M. (1973): Linguistics and Information Science, Academic Press, New York.

Tulving, E. (1972): "Episodic and Semantic Memory", Organization of Memory, Tulving, E. & Donaldson, W. (eds.), Academic Press, New York, 382-404.

Weiss, S. (1974): "A System for Model-Based Computer-Aided Diagnosis and Therapy", Technical Report CBM-TR-27, Comp, Sci. Dept., Rutgers U..

Wilson, K. V. (1977): From Association to Structure, (to appear).

Winograd, T. (1972): Understanding Natural Language, Academic Press, New York.

Winograd, T. (1975): "Frame Representations and the Declarative/Procedural Contraversy", Representation and Understanding, Bobrow D. & Collins, A. (eds.), Academic Press, New York, 185-210.

Woods, W. A. (1975): "What's in a Link: Foundations for Semantic Networks", Representation and Understanding, Bobrow, D. & Collins, A. (eds.), Academic Press, New York, 35-82.

Woods, W., Kaplan, R. & Nash-Webber, B. (1972): "The Lunar sciences natural language information system: final report", Technical Report 2378, Bolt, Beranek & Newman, Cambridge, Massachusetts.

Zadeh, L. (1965): "Fuzzy Sets", Information and Control 8, 338-353.

Zadeh, L. (1974): "The concept of a linguistic variable and its application to approximate reasoning", Learning Systems and Intelligent Robots, Fu, K. & Tou, J. (eds.), Plenum Press, New York, 1-10.

Zadeh, L. (1977): "PRUF - A Language for the Representation of Meaning in Natural Language", Proc. IJCAI 5, Cambridge, Massachusetts, August, 918.

Appendix 1

Examples of Combined Credibility Distributions

The following table summarizes the combination of credibility distributions for the given distributions D1 and D2. The rules in Chapter two, Section 2.2.1.5 were used to derive the combined distribution values.

SUMMARY

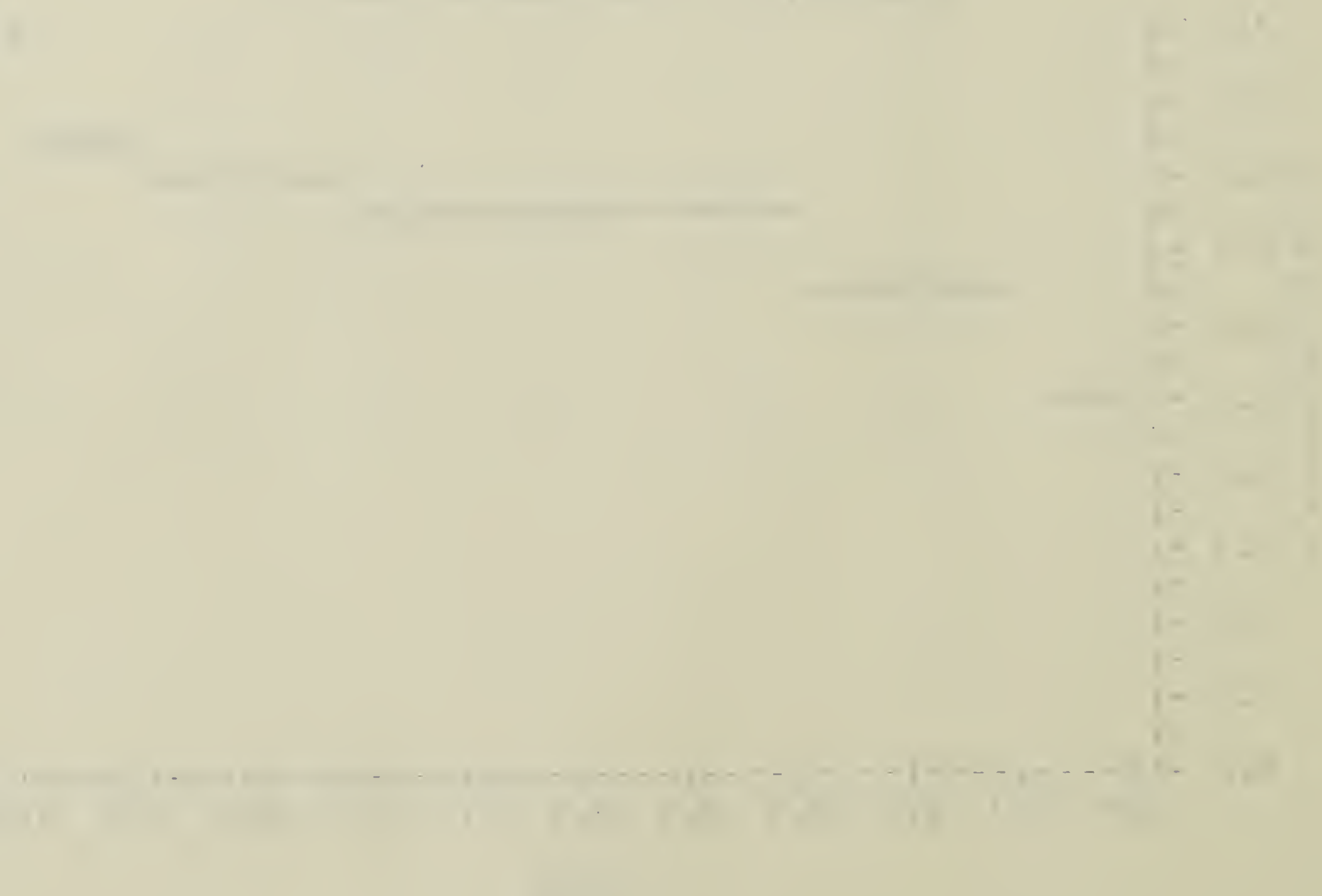
P(T) <=	0.0	0.1	0.3	0.5	0.7	0.9	1.0
D1	0.0	0.2	0.4	0.6	0.8	0.9	1.0
D2	0.5	0.65	0.75	0.75	0.8	0.85	1.0
¬D1	0.1	0.2	0.4	0.6	0.8	1.0	1.0
¬D2	0.15	0.2	0.25	0.25	0.35	0.5	1.0
D1 & D2	0.5	0.72	0.85	0.9	0.96	0.98	1.0
D1 | D2	0.0	0.13	0.3	0.45	0.64	0.76	1.0

```
                Credibility distribution for D1
  1.0 -|                                                  *
      -|
  0.9 -|                                             *****
      -|
C 0.8 -|                                   **********
r     -|
e 0.7 -|
d     -|
i 0.6 -|                         **********
b     -|
i 0.5 -|
l     -|
i 0.4 -|               **********
t     -|
y 0.3 -|
      -|
  0.2 -|     **********
      -|
  0.1 -|
      -|
  0.0 -+|----|----|----|----|----|----|----|----|----|----|
       0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0

                              Truth
```

```
                Credibility distribution for D2
  1.0 -|                                                  *
      -|
  0.9 -|
      -|                                             *****
C 0.8 -|                                   **********
r     -|               ********************
e 0.7 -|
d     -|     **********
i 0.6 -|
b     -|
i 0.5 -|*****
l     -|
i 0.4 -|
t     -|
y 0.3 -|
      -|
  0.2 -|
      -|
  0.1 -|
      -|
  0.0 -+|----|----|----|----|----|----|----|----|----|----|
       0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0

                              Truth
```

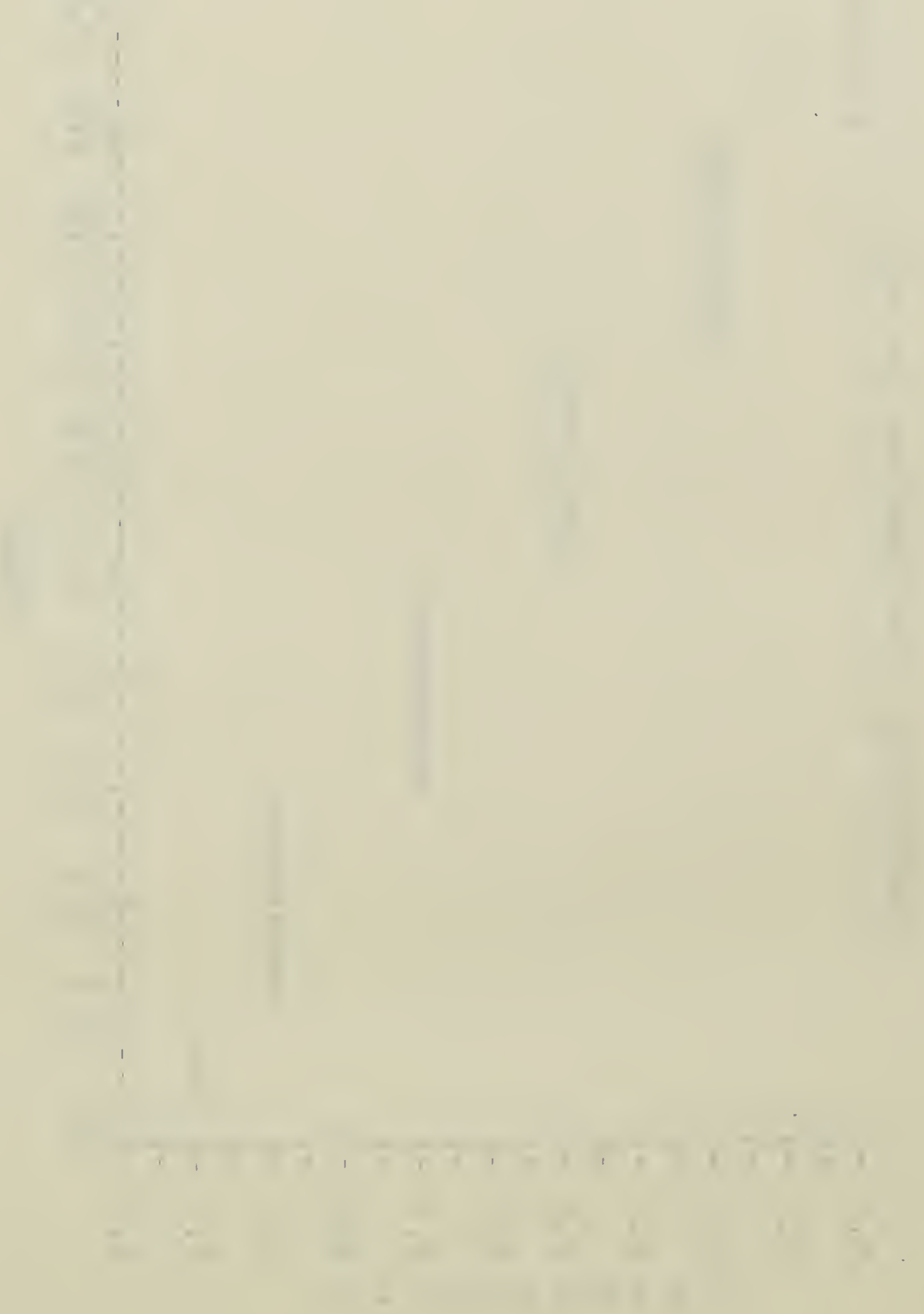


```
                Credibility distribution for ¬D1
  1.0 -|                                             ******
      -|
  0.9 -|
      -|
C 0.8 -|                                   **********
r     -|
e 0.7 -|
d     -|
i 0.6 -|                         **********
b     -|
i 0.5 -|
l     -|
i 0.4 -|               **********
t     -|
y 0.3 -|
      -|
  0.2 -|     **********
      -|
  0.1 -|*****
      -|
  0.0 -+|----|----|----|----|----|----|----|----|----|----|
       0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0

                               Truth
```

```
                Credibility distribution for ¬D2
  1.0 -|                                                  *
      -|
  0.9 -|
      -|
C 0.8 -|
r     -|
e 0.7 -|
d     -|
i 0.6 -|
b     -|
i 0.5 -|                                             *****
l     -|
i 0.4 -|
t     -|                                   **********
y 0.3 -|
      -|               ********************
  0.2 -|     **********
      -|*****
  0.1 -|
      -|
  0.0 -+|----|----|----|----|----|----|----|----|----|----|
       0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0

                               Truth
```



```
                Credibility distribution for D1 & D2
  1.0 -|                                                  *
      -|                                   ***************
  0.9 -|
      -|                         **********
C 0.8 -|               **********
r     -|
e 0.7 -|     **********
d     -|
i 0.6 -|
b     -|
i 0.5 -|*****
l     -|
i 0.4 -|
t     -|
y 0.3 -|
      -|
  0.2 -|
      -|
  0.1 -|
      -|
  0.0 -+|----|----|----|----|----|----|----|----|----|----|
       0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0

                               Truth
```

```
                Credibility distribution for D1 | D2
  1.0 -|                                                  *
      -|
  0.9 -|
      -|
C 0.8 -|
r     -|                                             *****
e 0.7 -|
d     -|
i 0.6 -|                                   **********
b     -|
i 0.5 -|
l     -|                         **********
i 0.4 -|
t     -|
y 0.3 -|               **********
      -|
  0.2 -|
      -|
  0.1 -|     **********
      -|
  0.0 -+|----|----|----|----|----|----|----|----|----|----|
       0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0

                               Truth
```

Appendix 2

PDB Input Grammar

The terminal STRING can be any alphanumeric string made up of the characters A - Z, a - z, -, 0 - 9. For example, the following are legal STRINGs:

STRING
SUPER-concept17
Fred-the-dog-who-lives-on-my-desk
PDB-CONCEPT-002758
bruce

```
statement          := function
                      prop
                      query
                      definition
                      topicquery

query              := prop?

prop               := quant-grp [ prop-body ]
                      quant-grp [ logop prop mod-grp ]
                      quant-grp [ prop impl prop mod-grp ]
                      quant-grp [ disjunct mod-grp ]
                      quant-grp [ conjunct mod-grp ]
                      label-use

topicquery         := STRING; STRING

definition         := { STRING deftype stringlist }

deftype            := SUBTOP
                      SUPTOP

stringlist         := STRING
                      stringlist STRING

conjunct           := prop and prop
                      conjunct and prop

disjunct           := prop or prop
                      disjunct or prop

function           := function-body )

prop-body          := predicate-grp mod-grp

function-body      := function-grp
```

```
                     function-grp truth-dist

predicate-grp     := node predicate
                     node predicate node
                     node predicate node node

function-grp      := function-node
                     function-node node
                     function-node node node

predicate         := STRING
                     STRING time-specifier

function-node     := ( predicate

node              := STRING
                     function
                     prop

mod-grp           :=                  /* optional field */
                     truth-dist
                     label-assignment
                     truth-dist label-assignment

truth-dist        := T*

quant-grp         :=                  /* optional field */
                     quant-grp quant

quant             := A* STRING
                     E* STRING

time-specifier    := < node >
                     < node , node >

logop             := not
                     nec

and               := &

or                :=  |

impl              := =>

not               := ¬

nec               := □
                     #

label-use         := :STRING

label-assignment  := =STRING
```

Appendix 3

PDB Program Examples

A prototype PDB model has been programmed to demonstrate the 'programmability' of the topic hierarchy organization. The system was programmed in the 'C' programming language, and runs under the UNIX operating system on a PDB 11/45 with 64K bytes of mainstore. The input parser was written with the aid of the 'YACC' compiler-compiler also available under the UNIX system.

The PDB program can maintain any number of fixed-size data bases, each consisting of a semantic network of propositions and an optional superimposed topic hierarchy structure. Each individual data base is kept off-line, and must be restored when in actual use. The input parser accepts all inputs constructed according to the grammar given in Appendix 2, although no further action is taken for function specifications or general propositional queries since neither a function evaluator nor a proposition matcher has been programmed.

Procedures for defining and superimposing a topic hierarchy organization on the PDB have been programmed, and an example of their operation is given below. Classification is done by a focus-finder routine, an augment-topic-access-skeleton routine, and an assert-proposition-in-category routine.

Cumulative credibility distributions may be attached to propositions and topic definitions, but as yet, the distributions are not utilized by the system.

The prototype model does little more than demonstrate how a topic hierarchy organization could operate. Internally, topic structures are represented as linked lists of integer indices. In a large 'production' system this type of data structure would eventually become inefficient in terms of both time and storage. The further development of efficient computational structures for representing topic hierarchies would greatly benefit from some type of content addressable, or psuedo content addressable (i.e. hashed) software and hardware data structures.

Examples

The PDB session below is essentially unaltered, except for indicating the user input with bold-face type. Note that the cumulative distribution function Fp(1.0) equals 1.0 in all cases, therefore the system does not prompt the user for that value.

```
*enter pdb system.
*pdb empty.
*  input
*enter input mode.
*input clear.
*?  A*x[[x DOG] => E*yE*t[[y CAT] & [x CHASES<t> y]] T* =X]
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  .2
prob at 0.5?  .3
prob at 0.7?  .5
prob at 0.9?  .5
*PDB insert UNNODE0000
*PDB insert internal dictionary WORD0068->DOG
*PDB insert CONCEPT0000->DOG
*PDB insert EXNODE0001
*PDB insert internal dictionary WORD0005->CAT
*PDB insert CONCEPT0001->CAT
*PDB insert EXNODE0002
*PDB insert internal dictionary WORD0035->CHASES
*PDB insert CONCEPT0002->CHASES
*PDB insert PROP0000
*PDB insert PROP0001
*PDB insert PROP0002
*PDB insert PROP0003
*PDB insert PROP0004
*insert A*x[[x DOG] => E*yE*t[[y CAT] & [x CHASES<t> y]] =X]
*CONCEPT FOCI:  DOG  EXNODE0001  EXNODE0002  CHASES
*assert A*x[[x DOG] => E*yE*t[[y CAT] & [x CHASES<t> y]] =X]
*input clear.
*?  [[Fred DOG] & [Fred BELIEVES :X]]
*PDB insert EXNODE0003
*PDB insert internal dictionary WORD0082->Fred
*PDB insert internal dictionary WORD0047->BELIEVES
*PDB insert CONCEPT0003->BELIEVES
*PDB insert PROP0005
*PDB insert PROP0006
*PDB insert PROP0007
*PDB insert PROP0008
*insert [[Fred DOG] & [Fred BELIEVES :X]]
*CONCEPT FOCI:  Fred
*assert [[Fred DOG] & [Fred BELIEVES :X]]
*input clear.
*?  end
*exit input mode.
*  list
PROP0000       TIME 1
ARG1   EXNODE0001
PRED   CAT
PROP0001       TIME 1
ARG1   UNNODE0000
TIMEARG1       EXNODE0002
PRED   CHASES
ARG2   EXNODE0001
```

```
PROP0002          TIME 1
ARG1    UNNODE0000
PRED    DOG
PROP0006          TIME 2
ARG1    Fred
PRED    DOG
PROP0007          TIME 2
ARG1    Fred
PRED    BELIEVES
ARG2    PROP0005
PROP0004          TIME 1
CREDIBILITY        0.00 0.00 0.20 0.30 0.50 0.50 1.00
SCOPE   EXNODE0001   EXNODE0002
ANTECEDENT
        PROP0002
CONSEQUENT
        PROP0003
PROP0005          TIME 2
ANTECEDENT
        PROP0002
CONSEQUENT
        PROP0003
PROP0003          TIME 1
CONJUNCTS
        PROP0001
        PROP0000
PROP0008          TIME 2
CONJUNCTS
        PROP0006
        PROP0007
*  list 4
PROP0004          TIME 1
CREDIBILITY        0.00 0.00 0.20 0.30 0.50 0.50 1.00
SCOPE   EXNODE0001   EXNODE0002
ANTECEDENT
        PROP0002          TIME 1
        ARG1    UNNODE0000
        PRED    DOG
CONSEQUENT
        PROP0003          TIME 1
        CONJUNCTS
                PROP0001          TIME 1
                ARG1    UNNODE0000
                TIMEARG1          EXNODE0002
                PRED    CHASES
                ARG2    EXNODE0001
                PROP0000          TIME 1
                ARG1    EXNODE0001
                PRED    CAT

*  list 7
PROP0007          TIME 2
ARG1    Fred
PRED    BELIEVES
ARG2    PROP0005
```

```
*  save fred
*pdb saved in file->fred.pdb
*  empty
*pdb empty.
*  input
*enter input mode.
*input clear.
*?  {appearance SUPTOP colour, texture, pattern}
*PDB insert TOPIC->appearance
*PDB insert internal dictionary WORD0092->appearance
*PDB insert internal dictionary WORD0091->APPEARANCE
*PDB insert CONCEPT0000->APPEARANCE
*PDB insert TOPIC->colour
*PDB insert internal dictionary WORD0008->colour
*PDB insert internal dictionary WORD0007->COLOUR
*PDB insert CONCEPT0001->COLOUR
*PDB insert TOPIC->texture
*PDB insert internal dictionary WORD0086->texture
*PDB insert internal dictionary WORD0085->TEXTURE
*PDB insert CONCEPT0002->TEXTURE
*PDB insert TOPIC->pattern
*PDB insert internal dictionary WORD0077->pattern
*PDB insert internal dictionary WORD0076->PATTERN
*PDB insert CONCEPT0003->PATTERN
appearance SUPER-TOPIC-OF colour
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  0
prob at 0.7?  0
prob at 0.9?  0
appearance SUPER-TOPIC-OF texture
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  0
prob at 0.7?  0
prob at 0.9?  0
appearance SUPER-TOPIC-OF pattern
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  0
prob at 0.7?  .2
prob at 0.9?  .2
*input clear.
*?  {colour SUPTOP grey, dark, shiny}
*PDB insert TOPIC->grey
*PDB insert internal dictionary WORD0059->grey
*PDB insert internal dictionary WORD0058->GREY
*PDB insert CONCEPT0004->GREY
*PDB insert TOPIC->dark
```

```
*PDB insert internal dictionary WORD0074->dark
*PDB insert internal dictionary WORD0073->DARK
*PDB insert CONCEPT0005->DARK
*PDB insert TOPIC->shiny
*PDB insert internal dictionary WORD0057->shiny
*PDB insert internal dictionary WORD0056->SHINY
*PDB insert CONCEPT0006->SHINY
colour SUPER-TOPIC-OF grey
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  0
prob at 0.7?  0
prob at 0.9?  0
colour SUPER-TOPIC-OF dark
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  .2
prob at 0.7?  .2
prob at 0.9?  .3
colour SUPER-TOPIC-OF shiny
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  .2
prob at 0.5?  .3
prob at 0.7?  .4
prob at 0.9?  .4
*input clear.
*?  {texture SUPTOP rough, smooth, shiny}
*PDB insert TOPIC->rough
*PDB insert internal dictionary WORD0010->rough
*PDB insert internal dictionary WORD0009->ROUGH
*PDB insert CONCEPT0007->ROUGH
*PDB insert TOPIC->smooth
*PDB insert internal dictionary WORD0020->smooth
*PDB insert internal dictionary WORD0019->SMOOTH
*PDB insert CONCEPT0008->SMOOTH
texture SUPER-TOPIC-OF rough
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  0
prob at 0.7?  0
prob at 0.9?  0
texture SUPER-TOPIC-OF smooth
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  0
```

```
prob at 0.7?  0
prob at 0.9?  .1
texture SUPER-TOPIC-OF shiny
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  .1
prob at 0.5?  .2
prob at 0.7?  .3
prob at 0.9?  .3
*input clear.
*?  {pattern SUPTOP spotted, striped}
*PDB insert TOPIC->spotted
*PDB insert internal dictionary WORD0089->spotted
*PDB insert internal dictionary WORD0088->SPOTTED
*PDB insert CONCEPT0009->SPOTTED
*PDB insert TOPIC->striped
*PDB insert internal dictionary WORD0043->striped
*PDB insert internal dictionary WORD0042->STRIPED
*PDB insert CONCEPT0010->STRIPED
pattern SUPER-TOPIC-OF spotted
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  0
prob at 0.7?  0
prob at 0.9?  0
pattern SUPER-TOPIC-OF striped
Enter values for truth distribution:
prob at 0.0?  0
prob at 0.1?  0
prob at 0.3?  0
prob at 0.5?  0
prob at 0.7?  0
prob at 0.9?  0
*input clear.
*?  A*x[[x ELEPHANT] => [x GREY]]
*PDB insert UNNODE0000
*PDB insert internal dictionary WORD0090->ELEPHANT
*PDB insert CONCEPT0011->ELEPHANT
*PDB insert PROP0000
*PDB insert PROP0001
*PDB insert PROP0002
*insert A*x[[x ELEPHANT] => [x GREY]]
*CONCEPT FOCI:  ELEPHANT  GREY
*TRIGGER TOPICS:  grey
*assert A*x[[x ELEPHANT] => [x GREY]]
*input clear.
*?  [Clyde SHINY]
*PDB insert EXNODE0001
*PDB insert internal dictionary WORD0053->Clyde
*PDB insert PROP0003
*insert [Clyde SHINY]
*CONCEPT FOCI:  Clyde
```

```
*TRIGGER TOPICS:  shiny
*assert [Clyde SHINY]
*input clear.
*?  Clyde;colour
*TOPIC categories scanned: colour  grey  dark  shiny
*relevant propositions: PROP0003
*input clear.
*?  Clyde;appearance
*TOPIC categories scanned: appearance  colour  texture
       pattern  grey  dark  shiny  rough  smooth  spotted
       striped
*relevant propositions: PROP0003
*input clear.
*?  ELEPHANT;texture
*TOPIC categories scanned: texture  rough  smooth  shiny
*no relevant propositions
*input clear.
*?  ELEPHANT;appearance
*TOPIC categories scanned: appearance  colour  texture
       pattern  grey  dark  shiny  rough  smooth  spotted
       striped
*relevant propositions: PROP0002
*input clear.
*?  A*x[[[x TOE] & [x PART-OF Clyde]] => [x SHINY]]
*PDB insert UNNODE0002
*PDB insert internal dictionary WORD0062->TOE
*PDB insert CONCEPT0012->TOE
*PDB insert internal dictionary WORD0071->PART-OF
*PDB insert CONCEPT0013->PART-OF
*PDB insert PROP0004
*PDB insert PROP0005
*PDB insert PROP0006
*PDB insert PROP0007
*PDB insert PROP0008
*insert A*x[[[x TOE] & [x PART-OF Clyde]] => [x SHINY]]
*CONCEPT FOCI:  TOE  Clyde  PART-OF  SHINY
*TRIGGER TOPICS:  shiny
*assert A*x[[[x TOE] & [x PART-OF Clyde]] => [x SHINY]]
*input clear.
*?  Clyde;appearance
*TOPIC categories scanned: appearance  colour  texture
       pattern  grey  dark  shiny  rough  smooth  spotted
       striped
*relevant propositions: PROP0003  PROP0008
*input clear.
*?  end
*exit input mode.
*  list
PROP0000       TIME 5
ARG1   UNNODE0000
PRED   ELEPHANT
PROP0001       TIME 5
ARG1   UNNODE0000
PRED   GREY
PROP0003       TIME 6
```

```
ARG1   Clyde
PRED   SHINY
PROP0004        TIME 11
ARG1   UNNODE0002
PRED   TOE
PROP0005        TIME 11
ARG1   UNNODE0002
PRED   PART-OF
ARG2   Clyde
PROP0007        TIME 11
ARG1   UNNODE0002
PRED   SHINY
PROP0002        TIME 5
ANTECEDENT
       PROP0000
CONSEQUENT
       PROP0001
PROP0006        TIME 11
CONJUNCTS
       PROP0004
       PROP0005
PROP0008        TIME 11
ANTECEDENT
       PROP0006
CONSEQUENT
       PROP0007
*  stop
*exit pdb system.
```